



# Algorithm Theoretical Basis Document (ATBD) for the Conical-Scanning Microwave Imager/Sounder (CMIS) Environmental Data Records (EDRs)

## Volume 16: Science Code Documentation Part 2: Description of Modules

**Version 2.1 – 15 March 2001**

**Solicitation No. F04701-01-R-0500**

Submitted by:  
**Atmospheric and Environmental Research, Inc.**  
**131 Hartwell Avenue**  
**Lexington, MA 02421-3126**

With contributions by:  
**Jean-Luc Moncet, Alan Lipton, Xu Liu, Hélène Rieu-Isaacs, John Galantowicz, Ernesto Sendoya, Scott Zaccheo, Richard Lynch, Jennifer Hegarty, Sid Boukabara, Yuguang He**

Prepared for:  
**Boeing Satellite Systems**  
**919 Imperial Avenue**  
**El Segundo, CA 90245**

**AER Document P757-TR-I-ATBD-SCDOC-2-20010315**



## TABLE OF CONTENTS FOR VOLUME 16, PART 2

Core Module Retrieval.....	5
Include Files.....	10
ProfIndx.incl .....	11
crims.incl.....	12
database.incl.....	16
eof.incl .....	18
govHeader.incl .....	19
input_desc.incl .....	21
inputs.incl.....	23
io_files.incl.....	26
osdrv.com .....	29
osdrv_ir.com.....	31
osdrv_mw.com .....	32
Subroutines .....	34
CopyArray.....	35
addCmisNoise .....	37
angles .....	39
bt .....	40
chkges .....	42
dbdt .....	44
deviceNoise.....	46
dminv .....	48
dmprod .....	50
draddt .....	52
fpath_mw .....	56
gauss.....	58
getChiSq.....	60
getEDR_clim.....	62
getSDR.....	64
getSDR_hdr.....	66
getpar_mw.....	68
handle_err .....	70
highEmisTest .....	72
init_Channels .....	74
init_OSS.....	76
init_Sy .....	78
invrt2 .....	80
io_mw_noise .....	82
isx.....	84
lpsum.....	86
lvl_int .....	87
mad.....	89
map_geo2retr .....	93
map_retr2geo .....	95
minv .....	97
mx_eof2geo.....	99
mx_geo2eof.....	101
mxd .....	102
mxv .....	104
openSimStatFiles .....	107
osdrv_mw .....	109
ossin_mw .....	111
osrad_mw .....	113
osstc_mw.....	116
osstran_mw .....	118
out_for_SimStat .....	120
planck .....	122

post_pro .....	124
printInfo .....	126
printInputs .....	128
putscene .....	130
qc .....	133
readStandardInputs .....	135
retr .....	137
setCovBack .....	139
setIprof .....	141
set_MW_Invert .....	143
sfcIceTest .....	145
shiftNcdfRec .....	147
smxpy .....	149
sunang .....	151
svp .....	155
ur_rand .....	157
ur_ranf .....	159
ur_rangef .....	161
ur_ranset .....	163
vadd .....	165
var_nedt .....	167
vsub .....	169
wnplan .....	171
writeout_hdr .....	173
writl2 .....	175
xtrns .....	178
Atmospheric Vertical Moisture Profile .....	181
Atmospheric Vertical Temperature Profile .....	184
Cloud Base Height .....	188
Cloud Ice Water Path .....	191
Cloud Liquid Water .....	194
Ice Surface Temperature .....	197
Land Surface Temperature .....	201
Pressure Profile .....	204
Precipitable Water .....	207
Sea Ice Age/Fresh Water Ice Age .....	210
Soil Moisture .....	215
Snow Cover/Depth .....	219
Total Water Content .....	222
Vegetation Surface Type .....	225
Wind Speed/Wind Stress/Wind Direction .....	229
Sea Surface Temperature .....	229
20 km WIND SPEED RETRIEVAL CODE .....	238
WIND STRESS COMPUTATION .....	240
WIND VECTOR RETRIEVAL CODE .....	241
Upper Atmospheric Vertical Temperature Profile .....	243
Precipitation Type/Rate .....	245

# Core Module Retrieval

## Program:

cmis

## Description:

cmis generates core retrieval products from truth scene and simulation calculations.

## Syntax:

cmis

## Example Run:

```
bin/cmis <Data/global/retr/namelist.conf >Data/global/retr/log.{iCell}.asc
```

## Inputs:

Inputs	Description
Namelist.conf	Configuration file that contains the names of each individual file, such as test.1.in
Log.{iCell}.asc	Log file for each cascade level (specified by iCell).

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nPar = 125 ;  
nTime = 6 ;  
nId = 8 ;  
nProfiles = UNLIMITED ; // (20 currently)  
nIrCldPoints = 2 ;  
nIrClds = 2 ;  
TT_iteration = 9 ;
```

variables:

```
float profiles(nProfiles, nPar) ;  
    profiles:long_name = "set of profile vectors" ;  
char id(nProfiles, nId) ;  
    id:long_name = "Identification String Tag" ;  
float lat(nProfiles) ;  
    lat:long_name = "latitude values" ;  
    lat:units = "degrees north" ;  
float lon(nProfiles) ;  
    lon:long_name = "longitude values" ;
```

```

    lon:units = "degrees east" ;
int landType(nProfiles) ;
    landType:long_name = "topographic type (ocean/land/etc)" ;
float pland(nProfiles) ;
    pland:long_name = "% land (0..1)" ;
float height(nProfiles) ;
    height:long_name = "satellite altitude" ;
    height:units = "km" ;
float scanAngle(nProfiles) ;
    scanAngle:long_name = "satellite angle" ;
int date(nProfiles, nTime) ;
    date:long_name = "date and time" ;
    date:units = "years-months-days-hours-min-secs" ;
float rms(nProfiles, TT_iteration) ;
    rms:long_name = "root mean squared of residual" ;
    rms:units = "Kelvin" ;
float chisq(nProfiles, TT_iteration) ;
    chisq:long_name = "Chi-Squared." ;
    chisq:units = "TBD" ;
int iteration(nProfiles) ;
    iteration:long_name = "iteration for retrieval" ;
float noise(nProfiles) ;
    noise:long_name = "total noise" ;
    noise:units = "Kelvin" ;

// global attributes:
:CreationTime = "20010202      115105.890" ;
:Process = "run/cmis/test/retr.mw1.scene.1.nc" ;
:DataType = "Geophysical Space" ;
:Temperature = 0, 40 ;
:Tskin = 40, 1 ;
:SurfacePressure = 41, 1 ;
:H2o = 42, 40 ;
:MwCloud = 82, 3 ;
:MwEmissivity = 85, 40 ;
:IrCloud1 = 125, 0 ;
:IrCloud2 = 125, 0 ;
:IrEmissivity = 125, 0 ;
:IrReflectivity = 125, 0 ;
:Co2 = 125, 0 ;
:O3 = 125, 0 ;
:N2o = 125, 0 ;
:Co = 125, 0 ;
:Ch4 = 125, 0 ;
:standardPressureGrid = 0.1f, 0.2f, 0.5f, 1.f, 1.5f, 2.f, 3.f, 4.f, 5.f, 7.f, 10.f,
15.f, 20.f, 25.f, 30.f, 50.f, 60.f, 70.f, 85.f, 100.f, 115.f, 135.f, 150.f, 200.f, 250.f, 300.f,
350.f, 400.f, 430.f, 475.f, 500.f, 570.f, 620.f, 670.f, 700.f, 780.f, 850.f, 920.f, 950.f,
1000.f ;

```

```
:mwSurfaceFrequencies = 6.625f, 6.625f, 10.65f, 10.65f, 10.65f,  
18.7f, 18.7f, 18.7f, 18.7f, 18.7f, 23.8f, 23.8f, 36.5f, 36.5f, 36.5f, 89.f,  
166.f, 183.31f, 183.31f, 183.31f, 117.543f, 115.85f, 113.2f, 50.3f, 52.24f, 53.57f, 54.38f,  
54.905f, 55.49f, 56.66f, 59.38f, 59.94f, 60.3712f, 60.408f, 60.4202f, 60.5088f ;  
:mwSurfacePolarities = 0, 1, 0, 1, 4, 5, 0, 1, 2, 3, 4, 5, 0, 1, 0, 1, 2, 3, 0, 1,  
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
```

### **Descriptions:**

#### **Dimensions:**

nPar: total length of the output vector.  
nTime: length of a date/time variable.  
nId: length of a ID string.  
nProfiles: total number of output profile  
nIrCldPoints: reserved length.  
nIrClds: reserved length.  
TT\_iteration: upper limit of iteration number for convergence  
check in the retrieval.

#### **Variables:**

profiles: profile vectors.  
id: NOAA-88 or other text identification string.  
lat: latitude in degrees north.  
lon: longitude in degree east.  
landType: land type. It is an enumerated number:  
Land=0,enForest=1,ebForest=2,dnForest=3,dbForest=4,  
Forest=5,cShrub=6,oShrub=7,oSavannas=8,wSavannas=9,  
Grass=10,Wetlands=11,Crops=12,Urban=13,  
Vegetation=14,SnowIce=15,Barren=16,Water=17.  
pland: percentage of land. Values are between 0 and 1.  
height: satellite height in kilometer.  
scanAngle: satellite scan angle in degree.  
date: date and time in year, month, day, hour, minute, second  
format.  
rms: root mean squared of residual.  
chisq: chi-squared value.  
iteration: maximum number of iteration for retrieval.  
noise: total background noise in Kelvin.

#### **Global Attributes:**

CreationTime: date and time that this netcdf file generated.  
Process: the name of this output file and its path.  
DataType: Describes the vector, is in Geophysical Space or in  
EOF Space.

Temperature: atmospheric vertical temperature profile in Kelvin  
 The two numbers after equal sign are offset and size respectively. All the following parameters follow the same rule. For example, Currently, temperature at index 0 in data vector, and 40 values given starts on the standard pressure grid defined by “standardPressureGrid” global attributes.  
 Tskin: skin temperature in Kelvin. Tskin starts at index 40 and a single value.  
 SurfacePressure: surface pressure in millibar. SurfacePressure starts at index 41 in data vector and a single value.  
 H2o: moisture profile in g/kg. It starts at index 42 in data vector and 40 values given on standard pressure grid defined by “standardPressureGrid” global attribute.  
 MwCloud: 3 parameter cloud or 40 values cloud profile. 3 parameters are: Cloud top pressure in millibar, cloud thickness in millibar and amount of cloud water in kg/m<sup>2</sup>. 40 values profile in kg/m<sup>3</sup>.  
 MwCloud starts at index 82 in data vector and total 3 values.  
 MwEmissivity: microwave emissivities profile. Values are between 0 and 1. Size is equal to number of channels. For example, MwEmissivity starts at index 85 in data vector and 40 values are corresponding to 40 microwave channels.  
 IrCloud1: Reserved parameter.  
 IrCloud2: Reserved parameter.  
 IrEmissivity: Reserved parameter.  
 Co2: Reserved parameter.  
 o3: Reserved parameter.  
 N2o : Reserved parameter.  
 Co: Reserved parameter.  
 Ch4: Reserved parameter.  
 StandardPressureGrid: pressure grid in millibar.  
 mwSurfaceFrequencies: microwave frequencies in Ghz.  
 mwSurfacePolarities: microwave polarities. It is an enumerated number.  
 Vertical=0, Horzitonal=1,  
 Positive 45 degree=2,  
 Negative 45 degree=3,  
 Left Circle=4,  
 Right Cricle=5.

#### Current Vector Contents:

Name	Temperature	Tskin	SurfacePressure	H2o	MwCloud	MwEmissivity
Index	0-39	40	41	42-81	82-84	85-124

Unit	Kelvin	Kelvin	mbar	g/kg	mbar, kg/m <sup>2</sup>	None
------	--------	--------	------	------	-------------------------	------

Total length: 125

**Source Code Location:**

common/src

common/include

retr/src

retr/include

devices/cmis/src

devices/cmis/include/retr

**Executable Binary Code Location:**

bin/

## **Include Files**

# ProfIdx.incl

**File Name:**

common/include/ProfIdx.incl

**Purpose:**

Parameter indexing

**Usage:**

```
include "ProfIdx.incl"
```

**Description:**

This is where the vector concept is implemented for both geophysical and retrieval domains. All the parameters are bundled together and indexed between the geophysical and the retrieval domain. It provides an easy way for the manipulation of the vector and matrix for the inversion techniques described by Rodgers (1976). Notice that,

1. The parameters below are in the geophysical domain.  
The lengths are defined in the input\_desc.incl.
2. This file is shared by simulation and retrieval that have their own versions of input\_desc.incl.
3. All the indices are adjusted automatically once the lengths are determined. Unless the structure of the retrieval profile is changed the following assignments need not to be altered.

**Parameter Definition:**

Var	Description
---	-----
ITskinG	Skin temperature.

IPsfcG Surface pressure.  
IH2oG Water vapor.  
ICldMwG MW cloud liquid water.  
IEmMwG Surface MW emissivity.  
ICldIr1G IR cloud (type I) parameters.  
ICldIr2G IR cloud (type II) parameters.  
IEmIrG IR surface emissivity.  
IRef1G Surface solar reflectivity in the FWD model.  
ICo2G Carbon dioxide.  
IndO3 Ozone.  
IN2oG Nitrous gas.  
ICoG Carbon monoxide.  
ICh4G Methane.  
NParMwG number of elements in the MW profile.  
NParG number of elements in the IR profile.

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

## crims.incl

**File Name:**

common/include/crims.incl

**Purpose:**

Defines all the dimensions and some common blocks

**Usage:**

```
include "crims.incl"
```

**Description:**

This file is included for the major components of the OSS model. It has the upper limits for the number of channels, number of cloud formations, cloud hinge points, number of cloud retrievals, and number of FOVs. It also contains the common blocks for the frequency, emissivity and reflectivity indexing at the surface and cloud levels, and the hinge frequencies for the first guess.

**Parameter Definition:**

Var	Description
---	-----
mxChnCC	maximum number of channels used for cloud clearing
mxClust	maximum clusters with each FOR for cloud clearing
mxCldFm	maximum number of cloud formations
mxCldHgPts	maximum number of cloud hinge points
mxClds	maximum number of clouds the model can retrieve
Ninstr	number of sensors (MW + IR)
NGeoPar	number of geo-information (lat/lon, satellite height, scan angle) for the sensor
mxchir	maximum number of IR channels
mxchmw	maximum number of MW channels
mxchan	maximum number of MW+IR channels
mxfov	maximum number of FOVs
mIter	maximum number of iteration allowed in the retrieval
nemt	number of emissivity types
mxbak	maximum number of background types
nLandTypes	number of land types
nchan	actual total number of channels used
nchanmw	actual number of MW channels used

```

nchanir      actual number of IR channels used

frq          MW+IR frequency vector

indxSfcEmRf  index array for surface emissivity and reflectivity

alphaSfcEmRf index array for interpolation cofficients for the
               surface emissivity and reflectivity

indxCldEmRf  index array for cloud emissivity and reflectivity

alphaSfcEmRf index array for interpolation cofficients for the
               cloud emissivity and reflectivity

frqeg        hinge point frequency for the first guess emissivity

```

#### **Common Blocks:**

Name	Description
landTypeArr	specifies the land types
chdat	contains the number of channels and frequency array
sfc_hp	contains index and coefficient arrays for the                surface emissivity and reflectivity
cld_hp	contains index and coefficient arrays for the                cloud emissivity and reflectivity
sfc_hp_frq	contains the frequencies for the first guess                emissivity at the hinge points

#### **Includes:**

Name	Description
----	-----
ProfIndx.incl	defines the indices for the geophysical and                the retrieval domain

#### **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.



# database.incl

**File Name:**

retr/include/database.incl

**Purpose:**

Defines common blocks for background information

**Usage:**

```
include "database.incl"
```

**Description:**

The file defines the arrays for background profile covariance, covariance conversion matrices between the geophysical and the retrieval domains.

**Parameter Definition:**

Var_Name	Description
ut_mw_cov_all	transposed matrix of the MW covariance at all backgrounds
u_mw_all	MW conversion matrix (eigenvector-matrix) beween the geophysical and the retrieval domains
ut_mw_all	transposed MW conversion matrix beween the geophysical and the retrieval domains
dback0_all	MW background profiles
lastGeo	(not used for the current setup)

**Common Blocks:**

Name	Description
stdCovariances	contains the covariance matrices
stdEigenVectors	contains the eigenvector matrices
stdBackgrounds	contains the background profiles

oldDataBaseSettings (not used)

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by the Atmospheric and Environmental Research, Inc., 1999

# **eof.incl**

## **File Name:**

common/include/eof.incl

## **Purpose:**

Defines EOF conversion matrix and its common block

## **Usage:**

```
include "eof.incl"
```

## **Description:**

This file is included in the functions where EOF conversion is needed. It contains the conversion matrix of EOF.

## **Parameter Definition:**

Var_name	Description
umtx	EOF conversion matrix

## **Common Blocks:**

Name	Description
EOFcommon	contains EOF conversion matrix

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# govHeader.incl

**File Name:**

common/include/govHeader.incl

**Purpose:**

IPO EDR header parameters

**Usage:**

```
include "govHeader.incl"
```

**Description:**

The file specifies the parameters for the output  
of EDRs in the header of the IPO-formatted file.  
It is used in the function, writeout\_header.

**Parameter Definition:**

Var_Name	Description
-----	-----
NGov	number of levels in the EDR output
nemis	number of surface emissivity/reflectivity wavenumber hinge points
ncemis	number of cloud emissivity/reflectivity wavenumber hinge points
ncld	number of clouds layers
topog	topography parameter
iver	
ncldmx	maximum number of cloud layers
nemismx	maximum number of emissivity/reflectivity wavenumber hinge points
nspr	number of spare real values
nspi	number of spare integer values

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **input\_desc.incl**

**File Name:**

devices/cmis/include/retr/input\_desc.incl

**Purpose:**

Defines parameter lengths in the retrieval vector

**Usage:**

```
include "input_desc.incl"
```

**Description:**

Defines the parameter lengths for mapping  
the profiles for forward and the inversion processes,  
as well as the indexing in the inversion process.  
This file is used mainly by ProfIndx.incl, which  
is specified in the file common/include/.

The variables are specified according to  
the convention, "prefix"+"physical meaning"+"suffix",  
where prefix is either "N", for length, or "I", for  
index. The suffix is either with or without "G",  
which represent geophysical domain or retrieval  
domain, respectively. The physical meanings of the  
variables are defined as follows.

The dimensions are defined as,

NParMw=total number of elements in the MW retrieval vector

NPar=total number of elements in the IR retrieval vector

NLev=number of standard vertical levels

NLay=number of standard vertical layers

**Parameter Definition:**

Var_Name	Description
----------	-------------

-----	-----
-------	-------

Temp	temperature
Tskin	skin temperature
Psfc	surface pressure
H2o	water vapor
CldMw	MW cloud liquid water grids.
EmMw	MW emissivities
CldIr1	Cloud (type I) parameters in IR bands (top, fraction, amount, thickness, ...)
CldIr2	Cloud (type II) parameters in IR bands (top, fraction, amount, thickness, ...)
EmIr	hinge points of IR surface emissivity
Refl	hinge points of solar surface reflectance
Co2	carbon dioxide
O3	ozone
N2o	nitrous gas
Co	carbon monoxide
Ch4	methane

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# inputs.incl

**File Name:**

common/include/inputs.incl

**Purpose:**

Contains namelists and their common blocks

**Usage:**

```
include "inputs.incl"
```

**Description:**

The file specifies model I/O namelists and their common blocks by which the model is controlled. The file serves as the interface for the control of the model runs. It should be proceeded by crims.incl wherever it is used.

**Parameter Definition:**

Var_Name	Description
-----	-----
mxfov2	= (mxfov-1)*mxfov (mxfov: maximum number of FOVs)
mxiter	maximum iteration at the input for IR retrieval
nxiter	maximum iteration at the input for MW retrieval
kchanmw	channel selection flag for MW
ozofac_bg	scaling factor for the ozone background
nfor	number of FORs
chanFileName	channel flag file
scene file	scene (or profile) file
wptbProffile	WBTP profile file
wptbAuxfile	WBTP auxiliary file
wptbScanfile	WBTP scan file
irSDRfile	true/simulated IR SDR file

mw1SDRFile	true/simulated MW1 SDR file
mw2SDRFile	true/simulated MW2 SDR file
mw1RetrSDRFile	retrieved MW1 SDR file
mw2RetrSDRFile	retrieved MW2 SDR file
irRetrSDRFile	retrieved IR SDR file
mw1ccSDRFile	cloud-cleared MW1 SDR file
mw2ccSDRFile	cloud-cleared MW2 SDR file
ircSDRFile	cloud-cleared IR SDR file
mw1errSDRFile	error MW1 SDR file
mw2errSDRFile	error MW2 SDR file
irerrSDRFile	error IR SDR file
mwRetrNcFile	MW retrieved EDRs in netCDF format
irRetrNcFile	IR retrieved EDRs in netCDF format
guessRetrNcFile	first-guess EDRs in netCDF format
mwRetrAscFile	MW retrieved EDRs in IPO ASCII format
irRetrAscFile	IR retrieved EDRs in IPO ASCII format
qcAscFile	quality control file in ASCII format
retrMwRad	retrieved MW radiance in ASCII format

Namelist and common blocks:

Name	Description
stdInputs	contains nfov and nfor
stdIrInputs	contains maximum number of IR iteration and ozone scaling factor
stdMwInputs	contains MW channel information, maximum number of MW iteration, and land types
stdInputFiles	contains channel selection file
inputFiles	profile and its auxiliary files
outputFiles	SDR and retrieved EDR files

**Includes:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **io\_files.incl**

**File Name:**

common/include/io\_files.incl

**Purpose:**

Defines all the logic units, and the table data

**Usage:**

```
include "io_files.incl"
```

**Description:**

The file specifies model I/O namelists and their common blocks for the forward model channel coefficients and the optical depth tables. The file serves as the part of the interface for the control of the model runs.

**Parameter Definition:**

Var_Name	Description
-----	-----
Lnsir	unit for IR noise file
Lfq	unit for frequency file
LcvAtm	unit for ASCII atmospheric covariance file
LcvEm	unit for ASCII emissivity covariance file
Lch	unit for channel information file
Lnsmw	unit for MW noise file
Lem	unit for the first guess IR emissivity file
Lo3bg	unit for the ozone background file
Lsc	unit for channel selection file
Linf	unit for IR information file
Lcfs	unit for IR coefficient file
Lbgd	unit for IR layer-averaged temperature table

Lh2	unit for IR layer-averaged moisture table
Lhs	unit for IR layer-averaged fixed-gas table
Lo3	unit for IR layer-averaged ozone table
Lco2	unit for IR layer-averaged CO2 table
Ln2o	unit for IR layer-averaged N2O table
Lco	unit for IR layer-averaged CO table
Lch4	unit for IR layer-averaged CH4 table
Lam	unit for MW information and coefficient file
Lop	unit for MW optical depth table
Lmwt	unit for MW layer-averaged temperature and moisture and table
Ldm	(not used)
Lnedt	unit for NEDT file
Lnrf	unit for noise reduction factor file
Lmw1	unit for true MW1 SDR file
Lmw2	unit for true MW2 SDR file
Lir	unit for true IR SDR file
Lrmw1	unit for retrieved MW1 SDR file
Lrmw2	unit for retrieved MW2 SDR file
Lrir	unit for retrieved IR SDR file
Lcmw1	unit for cloud-cleared MW1 SDR file
Lcmw2	unit for cloud-cleared MW2 SDR file
Lcir	unit for cloud-cleared IR SDR file
Lemw1	unit for error MW1 SDR file
Lemw2	unit for error MW2 SDR file
Leir	unit for error IR SDR file
LwpbtbProf	unit for WBTP profile file
LwpbtbAux	unit for WBTP auxiliary file
LwpbtbScan	unit for WBTP scan file
LmwRetrAsc	unit for the IPO format EDR file at MW retrieval

LirRetrAsc	unit for the IPO format EDR file at IR retrieval
LqcAsc	unit for the quality control file
covarFile	covariance file
nednFile	NEDN file
freqFile	IR frequency file
chan_parms	IR channel parameter file
noiseParFile	IR noise parameter file
emisfilg	first guess emissivity file
mwrsk	MW information and coefficient file
mwopt	MW optical depth file
mwtbl	MW layer-averaged temperature and moisture file
nedtFile	NEDT file
noiseReductFactFile	noise reduction factor file

Namelists and common blocks:

Name	Description
sensorConf	contains sensor-associated input files
inputConf	contains model-associated input files

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# ossdrv.com

**File Name:**

common/include/ossdrv.com

**Purpose:**

Contains common blocks for trace gas selection and  
pressure vector

**Usage:**

```
include "ossdrv.incl"
```

**Description:**

This file is used for passing selection flags for trace gases  
and the pressure grid. The selection vector is initialized in  
setIprof.f and the pressure grid is defined in crims.f.

**Parameter Definition:**

Var_Name	Description
nmol	maximum number of trace gases
deg2rad	unit conversion factor from degree to radian
ImolG	vector containing starting indices in the retrieval vector for the trace gases
Iprof	selection vector for the trace gases
pref	pressure grid

**Common Blocks:**

Name	Description
molIndx	contains ImolG and Iprof
ossPGrid	contains pressure grid

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

This page intentionally left blank.

# **ossdrv\_ir.com**

## **File Name:**

common/include/ossdrv\_ir.com

## **Usage:**

```
include "ossdrv_ir.com"
```

Name	Description
coefIR	contains coef,isel,ncoefs,nsmp
chdatossIR	contains nchossir,frqossir
odfix	contains odfixi
abh2o	contains abh2oi
abslf	contains abslfi
abo3	contains abo3i
tmptabc	contains tmptab,wh2oref,wo3ref

Developed by the Atmospheric and Environmental Research, Inc., 1999

Copyright 1999, Atmospheric and Environmental Research, Inc., 1999

## **Description:**

define common blocks that are used by the oss ir model.

# **ossdrv\_mw.com**

## **File Name:**

common/include/ossdrv\_mw.com

## **Purpose:**

Defines common blocks used in OSS MW model

## **Usage:**

```
include "ossdrv_mw.com"
```

## **Description:**

The file contains the common blocks of MW channel information, coefficients, optical depth, and (integrated) column temperature and moisture. The file is used by all the major components of the MW forward model such as ossin\_mw, osstran\_mw, and ossrad\_mw.

## **Parameter Definition:**

Var_Name	Description
-----	-----
mxtmp	maximum number of layers for the column temperature
mxwvp	maximum number of layers for the column moisture
ncmax	maximum number of coefficients in each MW channel
mxsmp	maximum number of spectral sample points in the MW bands
coef	coefficient table in all the MW channels
isel	coefficient selection array for IR bands
ncoefs	vector for the number of coefficients in each MW channel

nsmp	total number spectral sample points in MW band
nchanossmw	actual number of MW channels
frqossmw	MW frequency vector
odfixi	optical depth of the fixed-gases
abh2oi	water vapor absorption coefficients
tmptab	tabulated column temperature
wvptab	tabulated column moisture

**Common Blocks:**

Name	Description
coefmw	contains coef, isel, ncoefs, nsmp
chdatossMW	contains nchossmw, frqossmw
odfixmw	contains odfixi
abh2omw	contains abh2oi
proftab	contains tmptab, wvptab

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by the Atmospheric and Environmental Research, Inc., 1999

# **Subroutines**

# CopyArray

**File Name:**

retr/src/retr.f

**Purpose:**

Copies an array

**Usage:**

```
call CopyArray(profin,profout,Length)
```

**Description:**

This function copies an old array to a new array.

**Inputs:**

Var_Name	Type	Description
-----	----	-----
profin	real	input array
length	real	array length

**Outputs:**

Var_Name	Type	Description
-----	----	-----
profout	real	output array with identical content

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.



# **addCmisNoise**

## **File Name:**

devices/cmis/src/addDeviceNoise.f

## **Purpose:**

Add device-specific noise to the brightness  
temperature in simulation.

## **Usage:**

```
call addDeviceNoise(Y, deviceNoise, kchan, icell)
```

## **Description:**

Add device-specific noise to the brightness  
temperature in simulation using gaussian  
function.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
Y	real	radiance array.
devNoise	real	device-specific noise array.
kchan	integer	channel on/off flag array.
icell	integer	index for cell size selection. (reference to control.conf.cmis).

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
Y	real	radiance array that overwrites the same input array.

## **Common Blocks:**

defined in the includes

**Includes:**

File_Name	Description
-----	-----
crims.incl	defines all the dimensions and common blocks for emissivity, reflectivity, and frequency

**Copyright:**

AER, Inc., 2000

Developed by Atmospheric and Environmental Research, Inc., 2000

# angles

## File Name:

retr/src/angles.f

## Purpose:

Calculate satellite view angle

## Usage:

```
call angles(ang,hite,zenang,path,smax,crh)
```

## Description:

Calculate satellite view parameters in cmis.

Routine to perform path geometry calculations for  
satellite viewing earth at nadir angle, satang

## Inputs:

Var_Name	Type	Description
ang	real	satellite nadir angle of view (unit: degrees)
hite	real	satellite height (unit: km)

## Outputs:

Var_Name	Type	Description
zenang	real	zenith angle of path at earth surface (unit: degrees)
path	real	air mass factor
smax	real	nadir angle of view to earth edge (unit: degrees)
crh	real	ratio of satellite orbital radius to earth radius

## Common Blocks:

none

## Includes:

none

**Externals:**

none

**Copyright:**

n/a

Developed by J. Eyer, 1988

## **bt**

**File Name:**

common/src/planck.f

**Purpose:**

Converts radiance to brightness temperature

**Usage:**

```
brightTemp = bt(vn, rad)
```

**Description:**

At specified frequency, the function converts  
the radiance (mw/m<sup>2</sup>/str/cm<sup>-1</sup>) to the brightness  
temperature (K) .

**Inputs:**

Var_name	Type	Description
vn	real	frequency (wave number)
rad	real	radiance (mw/m <sup>2</sup> /str/cm <sup>-1</sup> )

**Outputs:**

Var_name	Type	Description
----------	------	-------------

-----  
bt            real       brightness temperature (K)

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# chkges

## File Name:

retr/src/chkges.f

## Purpose:

Checks all variables within their physical limits

## Usage:

```
call chkges(prof,ifail,x,SxMw,SxMwCld,modeIR)
```

## Description:

The temperature is checked against the lower and upper extremes of the climatological observations.

The moisture is checked against the saturation condition. For the microwave cloud, there are two situations. One is when only three parameters are retrieved, i.e., cloud top, cloud thickness, and cloud amount. In this case, the covariances are justified whenever these parameters are out of physical allowances. The second case is when the cloud profile is retrieved. Currently, this functionality is not installed. For the infrared cloud, the checking is always for the three parameters, i.e., the cloud top, the cloud thickness, and the cloud fraction. The checking process is similar to what has been done to the microwave cloud, where the covariances are justified.

## Inputs:

Var_Name	Type	Description
----------	------	-------------

-----	-----	-----
prof	real	contains profiles in geophysical space.
x	real	contains profiles in retrieval space.
SxMw	real	MW covariance.
SxMwCld	real	MW cloud covariance.
mode	integer	checking mode.

**Outputs:**

Var_Name	Type	Description
-----	-----	-----
ifail	integer	flag for checking (0=ok,1=failed).

**Common Blocks:**

Name	Description
-----	-----
ossPgrid	passing pressure.

**Includes:**

Name	Description
-----	-----
crims.incl	dimension declaration.

**Externals:**

Name	Description
-----	-----
svp	saturation vapor pressure.

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **dbdt**

## **File Name:**

common/src/planck.f

## **Purpose:**

Calculates the derivative of radiance w.r.t. temperature

## **Usage:**

```
dRaddTemp = dbdt(vn, t)
```

## **Description:**

At specified frequency, the function calcualtes the derivative of the radiance (or brightness temperature) w.r.t. temperature (K). This is a double-precision function. Please refer to draddt for single-precision process.

## **Inputs:**

Var_name	Type	Description
vn	real*8	frequency (wave number)
t	real*8	temperature (K)

## **Outputs:**

Var_name	Type	Description
dbdt	real*8	derivative of brightness temperature w.r.t. temperature

## **Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# deviceNoise

## File Name:

devices/cmis/src/deviceNoise.f

## Purpose:

compute CMIS-associated noise.

## Usage:

```
call deviceNoise(icellres,tb,rerr,sum_rerr,kchan)
```

## Description:

compute CMIS-associated noise.

## Inputs:

Var_Name	Type	Description
icellres	integer	index for cell resolution.
tb	real	brightness temperature array.
kchan	integer	channel selection flag array.

## Outputs:

Var_Name	Type	Description
rerr	real	instrument noise array.
sum_rerr	real	summation of the instrument noise.

## Common Blocks:

defined in the includes

## Includes:

File_Name	Description
crims.incl	defines all the dimensions and common blocks for

emissivity, reflectivity, and frequency  
io\_files.incl defines all the OSS model table files and  
corresponding logic units

**Externals:**

none

**Copyright:**

AER, Inc., 2000.

Developed by Atmospheric and Environmental Research, Inc., 2000

# dminv

## File Name:

retr/src/dminv.f

## Purpose:

Inverts a double precision matrix

## Usage:

```
call dminv(a,n,d,l,m)
```

## Description:

Routine to invert a double precision matrix.

The standard gauss-jordan method is used.

the determinant is also calculated. a determinant  
of zero indicates that the matrix is singular.

## Inputs:

Var_name	Type	Description
-----	----	-----
a	real*8	input matrix (destroyed in computation and replaced by resultant inverse.)
n	integer*4	order of matrix a
l	real*8	work vector of length n
m	real*8	work vector of length n

## Outputs:

Var_name	Type	Description
-----	----	-----
a	real*8	resultant inverse matrix
d	real*8	resultant determinant

## Common Blocks:

none

**Includes:**

none

**Externals:**

none

**Copyright:**

n/a

Developed by J.R.Eyre

# **dmprod**

## **File Name:**

retr/src/dmprod.f

## **Purpose:**

Does matrix multiplication

## **Usage:**

```
call dmprod(a,b,f,ni,nj,nk)
```

## **Description:**

Standard function for matrix multiplication  
according to the formula,  $f = a \cdot b$

## **Inputs:**

Var_Name	Type	Description
a	real	matrix of ni by nj.
b	real	matrix of nj by nk.
ni, nj, nk	integer	dimensions of matrices.

## **Outputs:**

Var_Name	Type	Description
f	real	matrix of ni by nk

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **draddt**

## **File Name:**

common/src/planck.f

## **Purpose:**

Calculates the derivative of radiance w.r.t. temperature

## **Usage:**

```
dRaddTemp = draddt(vn, t)
```

## **Description:**

At specified frequency, the function calcualtes the derivative of the radiance (or brightness temperature) w.r.t. temperature (K). This is a single-precision function. Please refer to dbdt for dboule-precision process.

## **Inputs:**

Var_name	Type	Description
vn	real	frequency (wave number)
t	real	temperature (K)

## **Outputs:**

Var_name	Type	Description
draddt	real	derivative of radiance (or brightness temperature) w.r.t. temperature

## **Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.



This page intentionally left blank.

# fpath\_mw

## File Name:

common/src/fpath\_mw.f

## Purpose:

Calculates MW layer-mean temperatures and layer amounts of H<sub>2</sub>O.

## Usage:

```
call fpath_mw(xG,NSurf,PSfc,TSfc,tavl,wml,dtu,dtl,dwqu,dwql)
```

## Description:

This function calculates the average temperature and molecular amounts for all layers for given profiles of temperature and moisture. It also calculates the derivatives of tavl with respect to a change in the lower and upper boundary temperatures and the derivatives of the H<sub>2</sub>O layer amounts with respect to a change in the H<sub>2</sub>O mixing ratio at the layer boundaries. Layer amounts are in molec./cm\*\*2. This version is for plane parallel geometry. Integration assumes that T is linear in z (LnT linear in LnP) and LnQ linear in LnP. This subroutine expects water vapor mixing ratios in g/kg.

## Inputs:

Var_Name	Type	Description
-----	----	-----
xG	real	profile vector in the geophysical space
Nsurf	integer	surface pressure level index
Psfc	real	surface pressure
Tsfc	real	ground level air temperature

## Outputs:

Var_Name	Type	Description
----------	------	-------------

```

-----  -----  -----
tavl      real   density-weighted layer-mean temperature
wml       real   layer amounts for all species
dtu       real   factor for level-to-layer derivative calculation
              (upper level temperature)
dtl       real   factor for level-to-layer derivative calculation
              (lower level temperature)
dwqu     real   factor for level-to-layer derivative calculation
              (upper level moisture)
dwql     real   factor for level-to-layer derivative calculation
              (lower level moisture)

```

**Common Blocks:**

defined in the includes

**Includes:**

included in ossdrv\_mw.com and ossdrv.com

File_Name	Description
-----  -----	
crims.incl	defines various dimensions and parameters for the retrieval
ossdrv_mw.com	maximum # of OSS points for MW, common blocks for gas optical depths and reference profiles
ossdrv.com	common blocks for IR+MW (gas selection, pressure grid)

**Externals:**

Name	Description
-----  -----	
lpsum	log-p interpolation

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **gauss**

## **File Name:**

common/src/gauss.f

## **Purpose:**

Computes Gaussian distribution

## **Usage:**

```
call gauss(ix,s,am)
```

## **Description:**

Computes a normally distributed random number with a given mean and standard deviation.

Uses 12 uniform random numbers to compute normal random numbers by central limit theorem. The result is then adjusted to match the given mean and standard deviation.

The uniform random numbers computed within the subroutine are found by the power residue method.

## **Inputs:**

Var_Name	Type	Description
ix	integer	the seed. It must contain a non-zero integer number with nine or less digits on the first entry to gauss or each time a re-initialization of the random sequence is desired. It is automatically reset to zero inside gauss after the random number generator has been re-initialized.
s	real	The desired standard deviation of the normal distribution.

am            real        The desired mean of the normal distribution.

**Outputs:**

Var_name	Type	Description
-----	----	-----
guass	real	random number with Gaussian distribution

**Common Blocks:**

none

**Includes:**

none

**Externals:**

Name	Description
----	-----
ur_rand	machine-independent random number generator

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# getChiSq

## File Name:

retr/src/retr.f

## Purpose:

Calculates Chi\_Square for quality control

## Usage:

```
call getChiSq(Ym,Y,DevNoise,chiSq,rms,kchan,nchanin)
```

## Description:

Chi\_square is used as one of the quality control parameters. The function returns the normalized chi\_square as well as rms.

## Inputs:

Var_Name	I*n	Description
-----	---	-----
Ym	real	reference radiance
Y	real	radiance
DevNoise	real	error matrix
kchan	integer	channel selection flag.
nchanin	integer	number of channels.

## Outputs:

Var_Name	I*n	Description
-----	---	-----
chiSq	real	chi_square
rms	real	root mean square of radiance difference.

## Common Blocks:

none

## Includes:

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **getEDR\_clim**

## **File Name:**

`retr/src/getEDR_clim.f`

## **Purpose:**

Loads the climatological background information

## **Usage:**

```
call getEDR_clim(itype)
```

## **Description:**

Opens, reads, and closes EDR means, error covariance,  
eigenvectors in netCDF format.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
itype	real	Input arg specifying the coded land type.

## **Outputs:**

none

## **Common Blocks:**

defined in `database.incl`

## **Includes:**

Name	Description
----	-----
<code>database.incl</code>	defines MW and IR covariance and background profiles and their common blocks
<code>crims.incl</code>	defines dimensions for the retroeval process
<code>netcdf.inc</code>	header file for netCDF functions

**Externals:**

netCDF standard FORTRAN function library

**Copyright:**

AER, Inc., 1999

Developed by AER, Inc., 1999

# getSDR

## File Name:

retr/src/getSDR.f

## Purpose:

Reads in SDRs (Sensor Data Records).

## Usage:

```
call getSDR(SdrFlg,FovFlg,NChan,Xid,GeoIn,TimeIn,y,Fail)
```

## Description:

A SDR file includes radiance, latitude/longitude, date, satellite information, measurement IDs. The function is set up to read SDRs either from the blind test files or the real measurements. The current data input format is IPO ASCII. The channel frequency includes MW1, MW2, and IR.

## Inputs:

Var_Name	I*n	Description
-----	----	-----
SdrFlg	integer	Vector of order 23 containing the checking flags for the selection among MW1 (1-15), MW2 (16-20), IR (20-23) bands.
FovFlg	integer	Vector of order mxfov containing the checking flags for all FOVs.
NChan	integer	Vector for Number channels of order 3.

## Outputs:

Var_Name	I*n	Description
-----	----	-----

ScanId	character*8	Scan ID of size 9 by 3.
GeoIn	real	Input geometric info of size 4 by 9 by 3 contains latitude, longitude, satellite altitude, satellite scan angle, for all the FOVs of all the instruments.
TimeIn	integer	Input date and time of size 6, which specified year, month, day, hour, minute, second.
y	real	Measured radiances (MW1,MW2,IR) of size (total) NChan by mxfov
Fail	integer	Flag to indicate the whether the input radiance is failed.

#### **Common Blocks:**

defined in crims.incl, inputs.incl

#### **Includes:**

Name	Description
----	-----
crims.incl	specifies the variable dimensions.
io_files.incl	defines the logical units of lookup tables for retrieval and their common blocks.
inputs.incl	defines I/O file names and their common blocks

#### **Externals:**

none

#### **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **getSDR\_hdr**

## **File Name:**

retr/src/getSDR\_hdr.f

## **Purpose:**

Reads the main header of a SDR file

## **Usage:**

```
call getSDR_hdr(NFOR,mw1SDRfile,mw2SDRfile,irSDRfile)
```

## **Description:**

Reads the main header of a SDR file. The SDR header contains the information of number of FORs, number of FOVs for specified scans and orbits. It also contains the standard frequency information. The information is passed out by common blocks as defined in the includes.

## **Inputs:**

Var_Name	Type	Description
mw1SDRfile	char*200	SDR file name for MW1
mw2SDRfile	char*200	SDR file name for MW2
irSDRfile	char*200	SDR file name for IR

## **Outputs:**

Var_Name	Type	Description
NFOR	integer	number of FORs

## **Common Blocks:**

defined in the includes

**Includes:**

Filename	Description
-----	-----
crims.incl	Specifies the dimensions for the retrieval process
io_files.incl	defines the logical units of lookup tables for retrieval and their common blocks.

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **getpar\_mw**

## **File Name:**

common/src/getpar\_mw.f

## **Purpose:**

Loads MW frequency

## **Usage:**

```
call getpar_mw(nchanmw)
```

## **Description:**

Opens and reads in MW-channel frequencies

## **Inputs:**

none

## **Outputs:**

Var_name	Type	Description
-----	----	-----
koffset	integer	number of the MW-channel frequencies

## **Common Blocks:**

defined in the includes

## **Includes:**

Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
io_files.incl	defines the I/O file paths
ossdrv_mw.com	maximum # of OSS points for MW, common blocks for gas optical depths and reference profiles

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc., 1997

# **handle\_err**

## **File Name:**

retr/src/getEDR\_clim.f

## **Purpose:**

netCDF file error handler

## **Usage:**

```
call handle_err(status)
```

## **Description:**

The function then calls the intrinsic function  
in the netCDF library to print out the error  
message corresponding to the input signal.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
status	integer	signal

## **Outputs:**

none

## **Common Blocks:**

none

## **Includes:**

Name	Description
----	-----
netcdf	defines the netCDF intrinsic function library

## **Externals:**

netCDF standard FORTRAN function library

## **Copyright:**

AER, Inc., 1999

Developed by AER, Inc., 1999

# highEmisTest

**File Name:**

devices/cmis/src/highEmisTest.f

**Purpose:**

Test brightness temperatures for signature of  
high-emissivity land surface

**Usage:**

```
call highEmisTest(Y, kchan, iHiEmFlag)
```

**Description:**

Test brightness temperatures for signature of  
high-emissivity land surface

**Inputs:**

Var_Name	Type	Description
---	---	-----
Y	real	radiance array.
kchan	integer	channel on/off flag array.

**Outputs:**

Var_Name	Type	Description
---	---	-----
iHiEmFlag	integer	flag is 1 if high emissivity land is detected and is 0 otherwise

**Includes:**

File_Name	Description
---	-----
crims.incl	defines various dimensions and parameters for the retrieval

ossdrv\_mw.com maximum #'s of OSS points for MW, common blocks for  
gas optical depths and reference profiles

**Copyright:**

AER, Inc., 2000

Developed by Atmospheric and Environmental Research, Inc.

# **init\_Channels**

## **File Name:**

common/src/initChSy.f

## **Purpose:**

Channel initialization

## **Usage:**

```
call init_Channels(kchanmw,kchan,chanFileName)
```

## **Description:**

By calling this routine with specified I/O variables (see below), the infrared and microwave channels (frequencies and total number of channels) are initialized by calling routines getpar\_mw and getpar\_ir (see the function definitions for detail). The routine also opens the channel selection file 'chanFileName' and reads in the channel selection specification, which makes channel selection an option.

## **Inputs:**

Var_name	Type	Description
chanFileName	character	File name for channel selection

## **Outputs:**

Var_name	Type	Description
kchanmw	integer	microwave channel selection flag
kchan	integer	infrared channel selection flag

## **Common Blocks:**

defined in the includes

**Includes:**

Name	Description
-----	-----
crims.incl	defines the dimensions for the variables and the common block for the MW+ IR frequency
io_files.incl	defines the I/O file paths

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc., 1999

# **init\_OSS**

## **File Name:**

common/src/initoss.f

## **Purpose:**

Initializes OSS model

## **Usage:**

```
call init_OSS(kchan)
```

## **Description:**

Loads the IR and MW optical depth tables, and computes  
IR radiance at the IR channels

## **Inputs:**

Var_name	Type	Description
-----	----	-----
kchan	integer	channel selection flag vector.

## **Outputs:**

none

## **Common Blocks:**

defined in the includes

## **Includes:**

Name	Description
----	-----
crims.incl	defines various dimensions and parameters for the retrieval
osssdrv_ir.com	contains the common block /sun_planck/

## **Externals:**

Name	Description
----	-----
ossin_ir	loads IR coefficients and optical depths

ossin\_mw      loads MW coefficients and optical depths

planck        planck function

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **init\_Sy**

## **File Name:**

common/src/initChSy.f

## **Purpose:**

Initialization of noise and error

## **Usage:**

```
call init_Sy(kchan, bnfreq, tsnois, rinois_arr,  
            rerr, sum_rerr, rerr_mw, iend)
```

## **Description:**

The routine opens IR sensor noise file and reads in frequencies, scene temperatures, and radiometric sensor noise for infrared bands. The routine also opens and reads in the error matrix for microwave sensors. All the file names and units opened are defined in the io\_files.incl file. The units are closed within the routine.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
kchan	integer	channel selection flag array

## **Outputs:**

Var_name	Type	Description
-----	----	-----
bnfreq	real	End-point frequencies for IR bands
tsnois	real	Scene temperature
rinois_arr	real	Radiometric noise at IR frequencies
rerr	real	Radiometric noise at all frequencies

rerr_mw	real	Radiometric noise at MW frequencies
sum_rerr	real	Summation of rerr_mw
iend	integer	End of file flag (0=ok, 1=eof)

**Common Blocks:**

defined in the includes

**Includes:**

Name	Description
-----	-----
crims.incl	defines the dimensions for the variables and the common block for the MW+ IR frequency
io_files.incl	defines the I/O file paths

**Externals:**

Name	Description
-----	-----
getpar_mw	loads MW frequency
getpar_ir	loads IR frequency
io_mw_noise	loads noise vector

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc., 1999

# **invrt2**

## **File Name:**

retr/src/invrt2.f

## **Purpose:**

Performs profile inversion.

## **Usage:**

```
call invrt2(xkt,e,c,ydif,xn1,xn,np,nchan)
```

## **Description:**

The inversion is performed by non-linear optimal estimation using a forecast profile and its error covariance as background constraints. Please refer to "CrIS ATBD" by AER in detail.

In order to accomodate the cray convention (row dimension = leading array dimension), the k matrix passed as an argument is stored as the transpose of the k matrix in xkt. The other matrices (e,c) are symmetrical. This version of invrt is to be used when the number of channels is small compared to the number of parameters.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
xkt	real	matrix to be inverted.
e	real	error covariance matrix.
c	real	profile covariance matrix.
ydif	real	difference of radiances.
xn1	real	initial profile.

```
np          real      number of parameters.  
nchan       real      number of channels.
```

### **Outputs:**

Var_Name	Type	Description
-----	----	-----
xn	real	inverted profile
SxMwOut	real	covariance matrix in EOF domain (used for cascade mode)

### **Common Blocks:**

none

### **Includes:**

none

### **Externals:**

Name	Description
----	-----
xtrns	transpose of a matrix.
dmprod	product of two matrices.
mad	summation of two matrices.
minv	inversion of a matrix.
mxv	product of a matrix and a vector.
smxpy	product of a vector and a matrix.

### **Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **io\_mw\_noise**

## **File Name:**

common/src/io\_mw\_noise.f

## **Purpose:**

Inputs noise vector

## **Usage:**

```
call io_mw_noise(rerr,fmerr,nc,io,iu,iend,kchanmw,  
sum_rerr,rerr_mw)
```

## **Description:**

I/O routine for tovs noise vectors.

Routine to input or output tovs noise vectors,  
radiometric (as rad) and forward model (as br.t)

## **Inputs:**

Var_name	Type	Description
-----	----	-----
rerr	real	radiometric noise vector, size nc (when io=0)
fmerr	real	forward model noise vector, size nc (when io=0)
nc	integer	length of vector (when io = 0)
io	integer	i/o flag: 0 for read, else for write
iu	integer	i/o unit number
iend	integer	eof flag: returned as 0=ok, 1=read eof

## **Outputs:**

Var_name	Type	Description
-----	----	-----
rerr	real	radiometric noise vector, size nc (when io=1)
fmerr	real	forward model noise vector, size nc (when io=1)
nc	integer	length of vector (when io=1)

```
rerr_mw      real      radiometric noise vector, same as rerr
```

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **isx**

## **File Name:**

retr/src/msub.f

## **Purpose:**

Performs transposing and negating matrix

## **Usage:**

```
call isx(a,b,n)
```

## **Description:**

The input matrix is transposed and negated first.

Then, the diagonal elements of the resultant matrix  
are incremented by 1.

## **Inputs:**

Var_name	Type	Description
a	real	a n by n matrix
n	integer	row and column length of the matrix

## **Outputs:**

Var_name	Type	Description
a	real	a n by n matrix

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# lpsum

## File Name:

common/src/fpath\_mw.f

## Purpose:

log-p interpolation.

## Usage:

```
call lpsum(pu,pl,xu,xl,scal,xint,dxu,dxl)
```

## Description:

The linear interpolation applies to the parameters  
in the pressure coordinate, and the linear relationship  
exists between the pressure and the parameter  
logarithmically.

## Inputs:

Var_Name	I*n	Description
pu	real	upper level pressure
pl	real	lower level pressure
xu	real	upper level variable
xl	real	lower level variable
scal	real	scaling factor

## Outputs:

Var_Name	I*n	Description
xint	real	interpolated variable
dxu	real	upper level conversion factor
dxl	real	lower level conversion factor

## Common Blocks:

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

## **lvl\_int**

**File Name:**

common/src/lvl\_int.f

**Purpose:**

Computes the surface value via interpolation

**Usage:**

```
call lvl_int(x,p,nx,plvl,xlvl,dxlvldp,il)
```

**Description:**

The interpolation of variables at the surface is done  
in the logarithmic pressure coordinate, given the pressure  
and the variable profiles, and the surface pressure.

**Inputs:**

Var_name	I*n	Description
-----	---	-----
x	real	profile to be interpolated.
p	real	pressure vector
nx	integer	dimension of x and p
plvl	real	surface pressure

**Outputs:**

Var_name	I*n	Description
-----	---	-----
xlvl	real	surface value obtained via interpolation
dxlvl dp	real	derivative of xlvl w.r.t. pressure
il	integer	index of surface pressure.

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **mad**

## **File Name:**

retr/src/msub.f

## **Purpose:**

Addition of a matrix and a vector

## **Usage:**

```
call mad(a,b,c,nr,nc)
```

## **Description:**

Calculates and returns the addition of a matrix and  
a vector

## **Inputs:**

Var_name	Type	Description
-----	----	-----
a	real	a nr by nc matrix
b	real	a vector vector
nr	integer	row length of the matrix
nc	integer	column length of the matrix or the length of the vector

## **Outputs:**

Var_name	Type	Description
-----	----	-----
c	real	product of the matrix and the vector

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.





# **map\_geo2retr**

## **File Name:**

retr/src/map\_geo2retr.f

## **Purpose:**

Converts the parameters from the geophysical to the retrieval domain.

## **Usage:**

```
call map_geo2retr(modeIr,xG,xBakG,Tsfc,x)
```

## **Description:**

Converts the parameters from the geophysical to the retrieval domain. The temperature and moisture are converted after being subtracted from the background values. It is the opposite process to map\_retr2geo.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
modeIr	integer	spared variable
xG	real	guess profile in the geophysical domain
xBakG	real	background profile in the geophysical domain

## **Outputs:**

Var_name	Type	Description
-----	----	-----
x	real	profile in the retrieval domain

## **Common Blocks:**

none

## **Includes:**

File_Name	Description
-----	-----

-----  
crims.incl define the dimension

**Externals:**

Name	Description
------	-------------

-----

mx_geo2eof	conversion from geophysical to retrieval space
------------	--

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **map\_retr2geo**

## **File Name:**

retr/src/map\_retr2geo.f

## **Purpose:**

Converts the retrieval parameters from the retrieval to  
the geophysical domain.

## **Usage:**

```
call map_retr2geo(modeIr, igeo, x, xBakG, xG, EmMw, xSfc, NSurf, EmRefl)
```

## **Description:**

Converts the retrieval parameters from the retrieval to  
the geophysical domain. The temperature and moisture are  
converted before being added with the background values.  
It is the opposite process to map\_geo2retr.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
modeIr	integer	converting mode
igeo	integer	MW geophysical scene flag
xBakG	real	background profile in the geophysical domain

## **Outputs:**

Var_name	Type	Description
-----	----	-----
x	real	profile in the retrieval domain
xG	real	guess profile in the geophysical domain
EmMw	real	emissivity for MW
Xsfc	real	skin temperature
Nsurf	integer	index corresponding to the surface pressure

EmRefl        real        spared variable

**Common Blocks:**

none

**Includes:**

File_Name	Description
-----	-----
crims.incl	define the dimension
ossdrv.com	include common blocks.
eof.incl	

**Externals:**

Name	Description
-----	-----
mx_eof2geo	conversion from retrieval to geophysical space.
lvl_int	interpolation of variables at the surface.
map_emis2instr_grid	interpolate the emissivity onto spectral points.

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **minv**

## **File Name:**

retr/src/dminv.f

## **Purpose:**

Inverts a double precision matrix

## **Usage:**

```
call minv(a,n,d,l,m)
```

## **Description:**

Routine to invert a double precision matrix.

The standard gauss-jordan method is used.

the determinant is also calculated. a determinant  
of zero indicates that the matrix is singular.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
a	real*8	input matrix (destroyed in computation and replaced by resultant inverse.)
n	integer*4	order of matrix a
l	real*8	work vector of length n
m	real*8	work vector of length n

## **Outputs:**

Var_name	Type	Description
-----	----	-----
a	real*8	resultant inverse matrix
d	real*8	resultant determinant

## **Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

n/a

Developed by J.R.Eyre

# **mx\_eof2geo**

## **File Name:**

retr/src/mx\_eof2geo.f

## **Purpose:**

Converts a matrix from the retrieval to the geophysical domain

## **Usage:**

```
call mx_eof2geo(mtxEof,mtxG,lenG,len,iwth)
```

## **Description:**

The function is used by map\_retr2geo to convert a matrix in the EOF (or retrieval) domain to the geophysical domain, after the inversion process. The length in the EOF domain is generally short than that in the geophysical domain, because of the application of eigenvectors.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
mtxEof	real	matrix of size len by iwth

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
mtxG	real	matrix of size lenG by nch to be converted.

## **Common Blocks:**

defined in the eof.incl

## **Includes:**

File_Name	Description
-----	-----
crims.incl	defines all the dimensions of the retrieval process

eof.incl            defines the common block for EOF inversion matrix

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **mx\_geo2eof**

## **File Name:**

retr/src/mx\_geo2eof.f

## **Purpose:**

Converts a matrix from the geophysical to the retrieval domain.

## **Usage:**

```
call mx_geo2eof(mtxG,mtxEOF,lenG,len,iwth)
```

## **Description:**

The function is used by map\_geo2retr to convert a matrix in the geophysical domain to the EOF (or retrieval) domain before the inversion process. The length in the EOF domain is generally short than that in the geophysical domain, because of the application of eigenvectors, which will make the inversion much faster.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
mtxG	real	matrix of size lenG by iwth

## **Outputs:**

Var_Name	Type	Description
-----	---	-----
mtxEOF	real	matrix of size len by iwth

## **Common Blocks:**

defined in the eof.incl

## **Includes:**

File_Name	Description
-----	-----

crims.incl define the dimension

eof.incl

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

## **mxd**

**File Name:**

retr/src/msub.f

**Purpose:**

Multiplies a matrix with a vector

**Usage:**

call mxd(a,b,c,nr,nc)

**Description:**

Calculates and returns the product of a matrix and  
a vector

**Inputs:**

Var_name	Type	Description
a	real	a nr by nc matrix
b	real	a vector vector
nr	integer	row length of the matrix
nc	integer	column length of the matrix or the length of the vector

**Outputs:**

Var_name	Type	Description
c	real	product of the matrix and the vector

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

## **mxv**

### **File Name:**

retr/src/invert2.f

### **Purpose:**

product of a matrix and a vector

### **Usage:**

```
call invert2(xkt,e,c,ydif,xn1,xn,np,nchan)
```

### **Description:**

A nar by nbr matrix is multiplied by a  
vector of length nbr to generate a vector  
of length nbr.

### **Inputs:**

Var_name	Type	Description
-----	----	-----
c	real	vector of length nar

### **Outputs:**

Var_Name	Type	Description
-----	----	-----
xn	real	inverted profile
SxMwOut	real	covariance matrix in EOF domain (used for cascade mode)

### **Common Blocks:**

none

### **Includes:**

none

### **Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.



# openSimStatFiles

**File Name:**

retr/src/retr.f

**Purpose:**

Opens files in IPO format

**Usage:**

```
call openSimStatFiles(mwOutFile,qcOutFile,  
                      mwOut,irOut,qcOut)
```

**Description:**

The function opens the MW and the quality control (QC) files.

**Inputs:**

Var_Name	Type	Description
-----	----	-----
mwOutFile	character*80	MW output file
qcOutFile	character*80	QC output file
mwout	integer	logic unit for MW
qcout	integer	logic unit for QC

**Outputs:**

none

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **ossdrv\_mw**

## **File Name:**

common/src/ossdrv\_mw.f

## **Purpose:**

main process for OSS forward model of MW band

## **Usage:**

```
call ossdrv_mw(xG,NSurf,Tsfc,EmMw,SatAng,y,xk,xkEmMw,kchan)
```

## **Description:**

Radiative transfer model for MW. It calls for all the OSS forward model functions for the calculation of radiance. It also calculates derivatives of radiances wrt profile parameters

## **Inputs:**

Var_Name	I*n	Description
---	---	-----
xG	real	profile vector in the geophysical space
Nsurf	integer	surface pressure level index
Tsfc	real	ground level air temperature
EmMw	real	MW emissivity
SatAng	real	satellite zenith angle
kchan	integer	channel selection flags.

## **Outputs:**

Var_Name	I*n	Description
---	---	-----
y	real	radiance vector
xk	real	derivatives of radiances wrt to model parameters
xkEmMw	real	derivatives of radiances wrt to MW surface

## emissivity

### Common Blocks:

defined in the includes

### Includes:

File_Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
ossdrv_mw.com	maximum #'s of OSS points for MW, common blocks for gas optical depths and reference profiles
ossdrv.com	defines common blocks for IR+MW

### Externals:

Name	Description
-----	-----
fpath_mw	average temperature and molecular amounts
osstran_mw	layer optical depths
osstc_mw	cloud optical depths for liquid clouds
ossrad_mw	brightness temperatures and derivatives of radiances

### Copyright:

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **ossin\_mw**

## **File Name:**

common/src/ossin\_mw.f

## **Purpose:**

Loads optical depth table for OSS MW model

## **Usage:**

```
call ossin_mw
```

## **Description:**

This is where all the coefficients and the optical depth tables in the MW band are loaded as one of the initialization processes.

## **Inputs:**

Var_name	I*n	Description
-----	---	-----
kchan	integer	channel selection flag vector.

## **Outputs:**

passed by common blocks

## **Common Blocks:**

Name	Description
----	-----
coefmw	contains coef, isel, ncoefs, nsmp
chdatossMW	contains nchossmw, frqossmw
odfixmw	contains odfixi
abh2omw	contains abh2oi
proftab	contains tmptab, wvptab

## **Includes:**

Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
osssdrv_mw.com	maximum #'s of OSS points for MW, common blocks for gas optical depths and reference profiles
io_files.incl	defines the I/O files and units

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **osssrad\_mw**

## **File Name:**

common/src/osssrad\_mw.f

## **Purpose:**

Calculates MW radiance and derivatives

## **Usage:**

```
call osssrad_mw(tautot,taucl,abscl,dtaucdtop,dtaucdthick,  
dtaudtmp,dtaudwvp,tavl,dtu,dtl,dwqu,dwql,NSurf,EmMw,xG,  
satang,imask,y,xk,xkEmMw)
```

## **Description:**

Compute brightness temperature (K) and their derivatives  
with respect to atmospheric and surface parameters.

The function also calculates transmittance before all  
the other process. The upward transmittance  
is calculated based upon the downward one.

## **Inputs:**

Var_Name	I*n	Description
-----	---	-----
tautot	real	total optical depth
taucl	real	optical depth of cloud
abscl	real	absorption coefficient for cloud liquid water
dtaucdtop	real	derivative of cloud optical depth w.r.t. cloud top
dtaucdthick	real	derivative of cloud optical depth w.r.t. cloud thickness
dtaudtmp	real	derivative of optical depth w.r.t. temperature.
dtaudwvp	real	derivative of optical depth w.r.t. water vapor
tavl	real	density-weighted layer-mean temperature

dtu	real	factor for level-to-layer derivative calculation (upper level temperature)
dtl	real	factor for level-to-layer derivative calculation (lower level temperature)
dwqu	real	factor for level-to-layer derivative calculation (upper level moisture)
dwql	real	factor for level-to-layer derivative calculation (lower level moisture)
imask	integer	channel selection flags
Nsurf	integer	index at surface pressure level
EmMw	real	MW emissivity
xG	real	profile vector in the geophysical space
SatAng	real	satellite zenith angle

#### **Outputs:**

Var_Name	I*n	Description
-----	---	-----
y	real	radiance vector
xk	real	derivatives of radiances wrt to model parameters
xkEmMw	real	derivatives of radiances wrt to MW surface emissivity

#### **Common Blocks:**

defined in crims.incl, ossdrv\_mw.com, and ossdrv.com

#### **Includes:**

File_Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
ossdrv_mw.com	maximum # of OSS points for MW, common blocks for gas optical depths and reference profiles
ossdrv.com	common blocks for IR+MW (gas selection, pressure grid)

#### **Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **osstc\_mw**

## **File Name:**

common/src/osstc\_mw.f

## **Purpose:**

Calculate MW cloud optical depths for liquid clouds

## **Usage:**

```
call osstc_mw(kchan,NSurf,Psfc,xG,tavl,taucl,abscl,  
              dtaucdtop,dtaucdthick)
```

## **Description:**

Based upon the features of the input cloud, i.e.,  
the cloud top, cloud thickness and amount, the  
function calculates MW cloud optical depths at  
each monochromatic channel for liquid cloud

## **Inputs:**

Var_Name	I*n	Description
-----	---	-----
kchan	integer	channel selection flags
Nsurf	integer	surface pressure level index
Psfc	real	surface pressure
xG	real	profile vector in the geophysical space
tavl	real	density-weighted layer-mean temperature

## **Outputs:**

Var_Name	I*n	Description
-----	---	-----
taucl	real	optical depth of cloud
abscl	real	absorption coefficient for cloud liquid water
dtaucdtop	real	derivative of cloud optical depth w.r.t. cloud top

```
dtaucdthick    real          derivative of cloud optical depth w.r.t. cloud  
                           thickness
```

**Common Blocks:**

defined in ossdrv\_mw.com and ossdrv.com

**Includes:**

File_Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
ossdrv_mw.com	maximum #'s of OSS points for MW, common blocks for gas optical depths and reference profiles
ossdrv.com	defines common blocks for IR+MW

**Externals:**

Name	Description
-----	-----
abcoefliq	computes absorption coefficient for suspended water droplets

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **osstran\_mw**

## **File Name:**

common/src/osstran\_mw.f

## **Purpose:**

Computes layer optical depths for molecular species  
in the MW band

## **Usage:**

```
call osstran_mw(tavl,wml,NSurf,tautot,dtaudtmp,dtaudwvp)
```

## **Description:**

Given the optical tables of each molecular species,  
the function computes its layer optical depths.  
Refer to ossrad\_mw for the transmittance and  
brightness temperature calculation

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
tavl	real	density-weighted layer-mean temperature
wml	real	molecular amounts for all species
Nsurf	integer	index at surface pressure level

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
tautot	real	total optical depth
dtaudtmp	real	derivative of optical depth w.r.t. temperature
dtaudwvp	real	derivative of optical depth w.r.t. water vapor
dtaucdtop	real	derivative of cloud optical depth w.r.t. cloud top

```
dtaucdthick    real          derivative of cloud optical depth w.r.t. cloud  
                           thickness
```

**Common Blocks:**

defined in ossdrv\_mw.com and ossdrv.com

**Includes:**

File_Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
ossdrv_mw.com	maximum #'s of OSS points for MW, common blocks for gas optical depths and reference profiles
ossdrv.com	defines common blocks for IR+MW

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **out\_for\_SimStat**

## **File Name:**

common/src/out\_for\_SimStat.f

## **Purpose:**

Outputs EDRs in the IPO ASCII format

## **Usage:**

```
call out_to_SimStat(GeoIn,TimeIn,ScanId,x,pref,Psfc,PLand,Lun)
```

## **Description:**

The function transforms the OSS retrieval profiles to the format generally used by the IPO. The reference routine is called SimStat.F in the IPO radiance simulation package.

## **Inputs:**

Var_Name	Type	Description
---	---	-----
GeoIn	real	Geometric infomation of size 4 by mxfov by 3 (latitude, longitude, satellite, altitude, scan angle, etc.)
TimeIn	integer	Vector of dynamic size, including date and time (yy,mm,dd,hh,mn,ss) .
ScanId	character*8	Scan ID of the MW1 footprint.
x	real	Vector of dynamic size, including all the true or retrieved parameters.
pref	real	Vector of dynamic size for the CrIMSS pressures.
Psfc	real	True surface pressure.
PLand	real	Percentage of Land (1=Land,0=water) .
iFlag	integer	QC control flag

```
fout      character*200 ilename for output.  
Lun       integer      ic unit number for fout.
```

### **Outputs:**

none

### **Common Blocks:**

none

### **Includes:**

Name	Description
-----	-----
crims.incl	defines various dimensions and parameters for the retrieval
govHeader.incl	defines parameters for Gov output header

### **Externals:**

Name	Description
----	-----
post_pro	interpolates OSS outputs according to the IPO format
writl2	writes routine according to IPO format

### **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# planck

**File Name:**

common/src/planck.f

**Purpose:**

Single-precision planck function

**Usage:**

```
plnk = planck(vn, tem)
```

**Description:**

At specified frequency, the function calcualtes the radiance at given temperature using the planck equation. Please refer to wnplan for the double-precision process.

**Inputs:**

Var_name	Type	Description
vn	real	frequency (wave number)
tem	real	temperature (K)

**Outputs:**

Var_name	Type	Description
planck	real	radiance (mw/m <sup>2</sup> /str/cm <sup>-1</sup> )

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **post\_pro**

## **File Name:**

common/src/post\_pro.f

## **Purpose:**

EDR post-processing

## **Usage:**

```
call post_pro(nlevi,pressi,tlevi,qlevi,sp,nlevo,presso,  
             tlevo,qlevo)
```

## **Description:**

Interpolates temperature and moisture from the model specified format to the reporting format. The interpolation is in the logarithmic pressure coordinate

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
nlevi	intger	Number of input pressure levels.
pressi	real	vector of pressure at the input levels.
tlevi	real	vector of temperature at input levels.
qlevi	real	vector of humidity at input levels.
sp	real	surface pressure.
presso	real	vector for output pressures.

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
tlevo	real	vector of temperature at output levels.
qlevo	real	vector of humidity at output levels.

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# printInfo

## File Name:

common/src/printInfo.f

## Purpose:

Print out EDR related information to standard output.

## Usage:

```
call printInfo(iFOR,iFOV,xid,sunZAng,scanAng,satAng,  
              satAlt,frq,Tsfc,Nsurf)
```

## Description:

Upon calling this module, all the EDR-related information is printed out to the standard output.

## Inputs:

Var	Type	Description
---	----	-----
iFOR	integer	FOR #
iFOV	integer	FOV #
xid	char*8	observation ID
x	real	observation vector
sunZAng	real	Solar zenith angle (deg)
scanAng	real	satellite scan angle (deg)
satAlt	real	Satellite altitude (km)
frq	real	MW frequency vector (GHz)
Tsfc	real	ground level air temperature
Nsurf	integer	surface pressure level index

## Outputs:

none

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc, 2000

Developed at Atmospheric and Environmental Research, Inc.

# **printInputs**

## **File Name:**

common/src/readStdInputs.f

## **Purpose:**

Prints standard inputs on screen

## **Usage:**

```
call printInputs()
```

## **Description:**

This function is mainly used by readStandardInputs  
to show the input arguments on screen.

## **Inputs:**

none

## **Outputs:**

none

## **Common Blocks:**

defined in the includes

## **Includes:**

File_Name	Description
-----	-----
crims.incl	defines all the dimensions and common blocks for emissivity, reflectivity, and frequency
inputs.incl	defines all the SDR and EDR I/O files

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1997



# **putscene**

## **File Name:**

retr/src/putscene.f

## **Purpose:**

Outputs scene profiles

## **Usage:**

```
call putscene(io_mode,xid,GeoIn,time,pland,x, fname,
             iter,rms,chisq,noise,ncid,stinfo,stprf,sttm,stid,
             strms,stchi)
```

## **Description:**

This function is used for all the processes (first guess and retrieval) for both microwave and infrared retrieval process to output the scene profiles in the netCDF format.

## **Inputs:**

Var_name	Type	Description
io_mode	integer	I/O mode. (0=open/create, 1=read, 2=write, 3=close)
ScanId	character	Profile ID. (e.g., IPO11111)
GeoIn	real	Geo-information, lat/lon, satellite altitude, satellite scan angle. (3 by 4)
time	integer	year-month-day-hour-minute-second
pland	real	Fraction of land.
x	real	Profile.
fname	character	Filename for output.
iter	integer	current iteration number

maxiter	integer	maximum iteration number
rms	real	r.m.s array for all the iterations
chisq	real	Chi-Square values for all the iterations
noise	real	standard noise
ncid	integer	netCDF file ID (or descriptor)
stinfo	integer	starting index for scalars
stprf	integer	starting indices for profile arrays
sttm	integer	starting indices for time vectors
stid	integer	starting indices for profile ID vectors
strms	integer	starting indices for the r.m.s arrays
stchi	integer	starting indices for Chi-square

### **Outputs:**

none

### **Common Blocks:**

defined in inputs.incl and ossdrv.com

### **Includes:**

Name	Description
----	-----
netcdf.inc	defines netCDF library functions
crims.incl	defines all the dimensions for the retrieval process
inputs.incl	defines I/O file names and paths
ossdrv.com	defines the common block for pressure levels

### **Externals:**

defined in the netcdf.inc

### **Copyright:**

1999, AER, Inc.

Developed by AER, Inc., 1999



# qc

## File Name:

retr/src/qc.f

## Purpose:

determine sc flag

## Usage:

```
call qc(pref,psfc,xGesG,Yref,Yges, summw, xGesGMw, rerr,  
       kchan,iflag)
```

## Description:

determine sc flag

## Inputs:

Var_Name	I*n	Description
-----	---	-----
pref	real	pressure profile (mb)
psfc	real	surface pressure (mb)
xGesG	real	second stage retrieval profile.
Yref	real	reference brightness temperature.
Yges	real	guess brightness temperature.
summw	real	total error.
rerr	real	radiative error of each channel.
xGesGMw	real	first stage retrieval profile.
pland	real	average land fraction in FOR.
kchan	integer	channel index.

## Outputs:

Var_Name	I*n	Description
-----	---	-----
iflag	integer	quality control flag.

**Common Blocks:**

none

**Includes:**

Name	Description
-----	-----
crims.incl	dimension declaration.

**Externals:**

none c\*

Developed by Atmospheric and Environmental Research, Inc.

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# readStandardInputs

**File Name:**

common/src/readStdInputs.f

**Purpose:**

Loads parameters from the namelist file

**Usage:**

```
call readStandardInputs
```

**Description:**

This is the first calling routine for the CrIMSS retrieval model. This routine requires a user input of a control file at runtime. By calling this routine, the control file is opened to input the runtime parameters by the namelists. The namelists are stdinputs, siminputs, stdinputfiles, and stdoutputfiles. The parameters are passed in to the model by common blocks with the same names (see below). The parameters are listed below. There are no explicit I/O variables in the function call. All the parameters are passed in via common blocks.

**Inputs:**

none

**Outputs:**

none

**Common Blocks:**

defined in the includes

**Includes:**

File_Name	Description
-----	-----
crims.incl	defines all the dimensions and common blocks for emissivity, reflectivity, and frequency
inputs.incl	defines all the SDR and EDR I/O files
io_files.incl	defines all the OSS model table files and corresponding logic units

**Externals:**

Name	Description
----	-----
printInput	prints on screen the inputs

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc., 1997

# **retr**

## **File Name:**

retr/src/retr.f

## **Purpose:**

main function for the retrieval process

## **Usage:**

retr

## **Description:**

program to retrieve geophysical parameters using  
OSS algorithms.

## **Inputs:**

Var_name	Description
-----	-----
[namelist]	file that contains the namelist for the I/O file paths

## **Outputs:**

Var_name	Description
-----	-----
xGesG	geophysical parameters of length NParG

## **Common Blocks:**

Described in ossdrv.com, eof.incl

## **Includes:**

Name	Description
----	-----
crims.incl	Defines all the dimensions of retrieval process

osssdrv.com      Defines the common blocks for the OSS model  
                  coefficients

inputs.incl     Defines the input file names and their common blocks

io\_files.incl    Defines all the logic units for the I/O files

database.incl    Defines covariance matrices and their common blocks

eof.incl        Defines the common block for the EOF matrix

**Externals:**

Name	Description
-----	-----

Copyright 1997, Atmospheric and Environmental Research, Inc.

Developed by Atmospheric and Environmental Research, Inc.

# setCovBack

## File Name:

retr/src/setCovBack.f

## Purpose:

Assign background covariance

## Usage:

```
call setCovBack(Ym0,plandin,ut_mw_cov,  
                 clw_cov_mw,dback0,igeo,nfov)
```

## Description:

The background and covariance are assigned  
in the function according to the land types.  
The land types and the surface pressure are  
loaded here as the auxiliary data.

## Inputs:

Var_name	I*n	Description
-----	---	-----
Ym0	real	radiance of size mxchan by mxfov
nfov	integer	number of FOVs
kchan	integer	channel on/off flag array.

## Outputs:

Var_name	I*n	Description
-----	---	-----
plandin	real	land fraction of various FOVs
ut_mw_cov	real	MW covariance
clw_cov_mw	real	MW cloud water covariance
dback0	real	background profile of size NparG
igeo	integer	geophysical info flag for MW

**Common Blocks:**

defined in includes.

**Includes:**

File_Name	Description
-----	-----
crims.incl	define the dimension
database.incl	
eof.incl	
io_files.inc	define the I/O units and file names.
inputs.incl	define the common block of landTypeArr.

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **setIprof**

## **File Name:**

common/src/setIprof.f

## **Purpose:**

Sets up flags for trace gas retrieval

## **Usage:**

```
call setIprof
```

## **Description:**

The function categorizes trace gas retrieval process in three cases: no retrieval, column retrieval, and profile retrieval. The trace gases include H<sub>2</sub>O, CO<sub>2</sub>, O<sub>3</sub>, N<sub>2</sub>O, CO, and CH<sub>4</sub>. The flags are passed via common blocks.

## **Inputs:**

none

## **Outputs:**

none

## **Common Blocks:**

defined in the includes

## **Includes:**

Name	Description
---	-----
crims.incl	defines various dimensions and parameters for the retrieval
osssdrv.com	defines the common block for ImolG and Iprof

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **set\_MW\_Invert**

## **File Name:**

retr/src/set\_mw\_invert.f

## **Purpose:**

Conversion of derivatives from geophysical to retrieval  
domain in the MW band

## **Usage:**

```
call set_MW_Invert(xG,Ym,Y,xkG,xkEmMw,rerr,xk,Sy,dely,nch,  
kchan,itermw,iGeo,Nsurf)
```

## **Description:**

The function converts the derivatives in the  
geophysical domain to the retrieval (or EOF)  
before the inversion process. The function also  
evaluates the error covariance matrix for the  
radiance in the MW band.

## **Inputs:**

Var_Name	I*n	Description
-----	---	-----
xG	real	profile in the geophysical space.
Ym	real	mean radiance.
Y	real	radiance.
xkG	real	derivative in the geophysical space.
xkEmMw	real	derivatives for MW emissivity
rerr	real	error matrix
kchan	integer	channel selection flags.
itermw	integer	iteration number for MW retrieval.
igeo	integer	flags for geophysical scenes.

Nsurf            integer        index at surface pressure level

**Outputs:**

Var_Name	I*n	Description
-----	---	-----
nch	integer	number of channels.
xk	real	derivative in the retrieval space.
Sy	real	error covariance matrix.
dely	real	difference b/w Ym and Y.

**Common Blocks:**

defined in the includes

**Includes:**

File_Name	Description
-----	-----
crims.incl	defines all the dimensions
eof.incl	defines the common block of EOF conversion matrix
osssdrv.com	defines common blocks for pressure

**Externals:**

Name	Description
----	-----
mx_ge2eof	conversion from geophysical into retrieval domain.

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# sfcIceTest

**File Name:**

```
devices/cmis/src/sfcIceTest.f
```

**Purpose:**

```
Test brightness temperatures for signature of  
sea ice or fresh water ice
```

**Usage:**

```
call sfcIceTest(Y, kchan, iSfcIceFlag)
```

**Description:**

```
Test brightness temperatures for signature of  
sea ice or fresh water ice
```

**Inputs:**

Var_Name	Type	Description
---	---	-----
Y	real	radiance array.
kchan	integer	channel on/off flag array.

**Outputs:**

Var_Name	Type	Description
---	---	-----
iSfcIceFlag	integer	flag is 1 if ice surface is detected and is 0 otherwise

**Includes:**

File_Name	Description
---	-----
crims.incl	defines various dimensions and parameters for the retrieval

ossdrv\_mw.com maximum #'s of OSS points for MW, common blocks for  
gas optical depths and reference profiles

**Copyright:**

AER, Inc., 2000

Developed by Atmospheric and Environmental Research, Inc.

# shiftNcdfRec

**File Name:**

retr/src/putscene.f

**Purpose:**

Shifts the record pointer

**Usage:**

```
call shiftNcdfRec(n1,n2,n3,n4,n5,n6)
```

**Description:**

In reading a netCDF file, the record may be accessed by shifting the record number. This function provides a sequential shifting mechanism for this purpose.

**Inputs:**

Var_name	Type	Description
n1 - n6	integer	record numbers

**Outputs:**

none

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

AER, Inc., 2000

Developed by AER, Inc., 2000

# **smxpy**

## **File Name:**

retr/src/invert2.f

## **Purpose:**

product of a vector and a matrix

## **Usage:**

```
call invert2(xkt,e,c,ydif,xn1,xn,np,nchan)
```

## **Description:**

A vector of length n2 is multiplied by a  
n2 by n1 matrix to generate a vector  
of length n1.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
y	real	vector of length nar

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
xn	real	inverted profile
SxMwOut	real	covariance matrix in EOF domain (used for cascade mode)

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **sunang**

## **File Name:**

common/src/sunang.f

## **Purpose:**

Calculates solar zenith angle

## **Usage:**

```
call sunang(mode,time,alat,alon,sundec,sunzang,dosun)
```

## **Description:**

The function evaluates the solar zenith angle by the given time of the day, latitude and longitude.

This routine is adapted from the pathfinder routine sunang.f. This is approximate, but adequate.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
mode	integer	0=solve for solar angle
time	integer	(year,month,day,hour,minute,second)

## **Outputs:**

Var_Name	Type	Description
-----	----	-----
sundec	real	sub-solar latitude.
sunzang	real	solar zenith angle
dosun	logic	T=include the solar term, F=otherwise.

## **Common Blocks:**

none

## **Includes:**

none

**Externals:**

none

**Copyright:**

n.a.

Modified by Atmospheric and Environmental Research, Inc., 1998.





# **svp**

## **File Name:**

retr/src/chkges.f

## **Purpose:**

Calculates saturation vapor pressure

## **Usage:**

vp = svp(t)

## **Description:**

Calculates saturated vapour pressure in mb  
at given temperature in K.  
Value corresponds to svp over water for  
temp 5 deg.C, to svp over ice for  
temp

## **Inputs:**

Var_name	Type	Description
-----	---	-----
t	real	temperature in K

## **Outputs:**

Var_name	Type	Description
-----	---	-----
svp	real	saturation vapor pressure (mb)

## **Common Blocks:**

Name	Description
----	-----
es_tbl	saturation vapor pressure table

## **Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **ur\_rand**

## **File Name:**

`./common/src/sysdependent.F`

## **Purpose:**

Generates random number

## **Usage:**

`x = ur_rand(iseed)`

## **Description:**

Generates random number between 0 and 1 of Gaussian distribution with the system dependency.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
iseed	integer	the seed for new sequence of random numbers.  If iseed = 0, the random number is within the same sequence as the last call.

## **Outputs:**

Var_name	Type	Description
-----	----	-----
x	real	random number between 0 and 1

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

Copyright, 1999, Atmospheric and Environmental Research, Inc.

Developed by the Atmospheric and Environmental Research, Inc., 1999

# **ur\_ranf**

## **File Name:**

`./common/src/sysdependent.F`

## **Purpose:**

Generates random number

## **Usage:**

`x = ur_ranf()`

## **Description:**

The function generates random number between 0 and 1  
of Gaussian distribution with the system dependency.  
The random number sequence should be initialized by  
the calling routine.

## **Inputs:**

none

## **Outputs:**

Var_name	Type	Description
<code>-----</code>	<code>-----</code>	<code>-----</code>
<code>x</code>	real	random number between 0 and 1

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

Copyright, 1999, Atmospheric and Environmental Research, Inc.

Developed by the Atmospheric and Environmental Research, Inc., 1999



# **ur\_ranget**

## **File Name:**

`./common/src/sysdependent.F`

## **Purpose:**

Gets a seed for random number sequence

## **Usage:**

`x = ur_ranget(iseed)`

## **Description:**

The function obtains a seed from the random number table.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
iseed	integer	on SUN machine, the number is used as the starting value and is replaced by adding a random number. On SGI, the number is replaced by machine on the way out.

## **Outputs:**

Var_name	Type	Description
-----	----	-----
x	integer*8	some dummy integer
iseed	integer*8	new seed

## **Common Blocks:**

none

## **Includes:**

none

**Externals:**

none

Developed by the Atmospheric and Environmental Research, Inc., 1999

Copyright, 1999, AER, Inc.

# **ur\_ranset**

## **File Name:**

`./common/src/sysdependent.F`

## **Purpose:**

Establishes a seed

## **Usage:**

`x = ur_ranset(iseed)`

## **Description:**

On SGI, the `ur_ranset` uses `ranset` function to establishes a seed in the random number seed table. There is no counter part on SUN. Therefore, the function simply returned the seed.

## **Inputs:**

Var_name	Type	Description
-----	----	-----
iseed	integer*8	new seed.

## **Outputs:**

Var_name	Type	Description
-----	----	-----
x	integer*8	some dummy integer

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

Copyright, 1999, Atmospheric and Environmental Research, Inc.

Developed by the Atmospheric and Environmental Research, Inc., 1999

# vadd

## File Name:

retr/src/msub.f

## Purpose:

Vector addition

## Usage:

```
call vadd(a,b,c,n)
```

## Description:

Calculates and returns the addition of the two vectors

## Inputs:

Var_name	Type	Description
-----	----	-----
a	real	first vector
b	real	second vector
n	integer	length of vectors

## Outputs:

Var_name	Type	Description
-----	----	-----
c	real	addition of the two input vectors

## Common Blocks:

none

## Includes:

none

## Externals:

none

## Copyright:

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **var\_nedt**

## **File Name:**

common/src/var\_nedt.f

## **Purpose:**

Compute nedt from input parameters

## **Usage:**

```
call var_nedt(ispec, tscene, time_set,  
               bw_set, ica_set,iun, nedt_file)
```

## **Description:**

compute nedt from input parameters used by  
deviceNoise.

## **Inputs:**

Var_Name	Type	Description
ispec	integer	identifying number of channel for data used.
tscene	real	scene temperature in Kelvin.
bw_set	real	bandwidth (Mhz), optional, if zero the tabulated value is used.
ica_set	integer	flag to indicate whether to use calibration amplification(1) or not(0).
iun	integer	unit number for file with tabulated NEDT parameters
nedt_file	character	name of file with tabulated NEDT parameters

## **Outputs:**

Var_Name	Type	Description
var_nedt	real	nedt (K)

## **Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **vsub**

## **File Name:**

retr/src/msub.f

## **Purpose:**

Vector subtraction

## **Usage:**

call vsub(a,b,c,n)

## **Description:**

Calculates and returns the difference between the  
two vectors

## **Inputs:**

Var_name	Type	Description
-----	----	-----
a	real	first vector
b	real	second vector
n	integer	length of vectors

## **Outputs:**

Var_name	Type	Description
-----	----	-----
c	real	difference of the two input vectors

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **wnplan**

## **File Name:**

common/src/planck.f

## **Purpose:**

Double-precision planck function

## **Usage:**

```
wn = wnplan(vn, tem)
```

## **Description:**

At specified frequency, the function calcualtes the radiance at given temperature using the planck equation. Please refer to planck for the single-precision process.

## **Inputs:**

Var_name	Type	Description
vn	real*8	frequency (wave number)
tem	real*8	temperature (K)

## **Outputs:**

Var_name	Type	Description
wnplan	real*8	radiance (mw/m <sup>2</sup> /str/cm <sup>-1</sup> )

## **Common Blocks:**

none

## **Includes:**

none

**Externals:**

none

**Copyright:**

Atmospheric and Environmental Research, Inc., 1997

Developed by Atmospheric and Environmental Research, Inc.

# **writeout\_hdr**

## **File Name:**

common/src/writeout\_hdr.f

## **Purpose:**

Outputs the main header in IPO format

## **Usage:**

```
call writeout_hdr(nobs,nemis,ncemis,  
nl2lev,ndescr,pobs,descr,iunit)
```

## **Description:**

The functions writes out the main header for the EDR retrieval. It is used jointly with writl2. The main header contains of the information of number of FOR, number of surface and cloud emissivity, pressure levels, and number of descriptors.

## **Inputs:**

Var_Name	Type	Description
-----	----	-----
nobs	integer	Number of Foot of Regard
nemis	integer	Number of surface emissivity
ncemis	integer	Number of cloud emissivity
nl2lev	integer	Number of pressure levels
ndescr	integer	Number of description lines in the header.
pobs	real	Vector of size nl2lev for pressures.
descr	character*80	Vector of size ndescr for the descriptions.
iunit	integer	Logic unit of "filename".

**Outputs:**

none

**Common Blocks:**

none

**Includes:**

Name	Description
govHeader.incl	defines parameters for Gov output header

**Externals:**

none

**Copyright:**

n/a

Developed by IPO and modified by AER, Inc., Feb., 1999

# writl2

## File Name:

common/src/writel2.f

## Purpose:

Outputs IPO formatted EDRs

## Usage:

```
call subroutine writl2(iver,  
&      iprt, iounit, unformatted, nl2lev, idprof,  
&      alat, alon, nyear, nmonth, nday, nhour, nminute, rsec,  
&      tair, h2ocd, ozocd, wlcd, ciw, ncld,  
&      pcldtop, cldfrc, cldemis, cldrho, cldfreq, ncemis,  
&      topog, pland, psurf, tsurf,  
&      emisir, rhoir, freqemis, nemis, emismw,  
&      nspr, rspare, nspi, ispare )
```

## Description:

The function writes to a file the IPO formatted EDRs. It is used jointly with the function writeout\_hdr. For the error handler, data is written for bad input data but error message is printed.

Data set format changes depending on the flag unformatted. If unformatted=.false., the subroutine writes a formatted data set. If unformatted=.true., the subroutine writes an unformatted data set. This makes I/O faster and saves disk space. Unfortunately, the unformatted data set is machine dependent.

**Inputs:**

Var_name	Type	Description
iprt	integer	flag for detailed print
iounit	integer	I/O unit for the level I data set
unformatted	logical	flag for using unformatted data set
nl2lev	integer	number of pressure levels
idprof	char*8	profile id
alat	real	latitude (degree)
alon	real	longitude (degree)
nyear	integer	year (e.g. 1900)
nmonth	integer	month of year (1 - 12)
nday	integer	day of month (1 - 31)
nhour	integer	hour of day (0 - 24)
nmin	integer	minute of hour (0 - 59)
rsec	real	second of minute (0.000 - 59.999)
tair	real	temperature profile (kelvin)
h2ocd	real	water vapor column density (mol/cm sq)
ozocd	real	ozone column density (mol/cm sq)
wlcd	real	water column density (mol/cm sq cloud/ice)
ciw	integer	switch (cloud=0/ice=1)
ncld	integer	number of cloud layers
pcldttop	real	cloud top pressure (millibar)
cldfrc	real	cloud fraction
cldemis	real	cloud IR emissivities
cldrho	real	cloud IR reflectance
cldfreq	real	frequencies for cloud emissivities (wave number)
ncemis	integer	number of cloud emissivity bands
topog	real	topography (meter)

pland	real	percentage of land surface
psurf	real	surface pressure (millibar)
emisir	real	IR surface emissivity
rhoir	real	IR surface reflectivity
freqemis	real	Frequencies for emissivities (wave number)
nemis	integer	number of IR emissivity bands
emismw	real	MW surface emissivities and reflectivities
rspare	real	real spares
ispare	integer	integer spares

**Outputs:**

none

**Common Blocks:**

none

**Includes:**

none

**Externals:**

none

**Copyright:**

n/a

Developed by IPO, 1991, and modified by AER, Inc., 1999

# **xtrns**

## **File Name:**

retr/src/msub.f

## **Purpose:**

matrix transpose

## **Usage:**

call xtrns(a,b,nr,nc)

## **Description:**

Tranposes a matrix

## **Inputs:**

Var_name	Type	Description
-----	----	-----
a	real	first vector
nr	integer	row length of the vector
nc	integer	column length of the vector

## **Outputs:**

Var_name	Type	Description
-----	----	-----
b	real	transpose of the input vectors

## **Common Blocks:**

none

## **Includes:**

none

## **Externals:**

none

## **Copyright:**

Atmospheric and Environmental Research, Inc., 1999

Developed by Atmospheric and Environmental Research, Inc.

# **EDRs**

# Atmospheric Vertical Moisture Profile

## Executable Program:

avmp

## Description :

avmp produces atmospheric vertical moisture profile at given pressure/altitude grids from core retrieval products

## Syntax:

**avmp [-a -C -h -i -I -o -O -r -W]**

## Options:

-a	append data to output file
-C	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name
-I <str>	input truth file name
-o <str>	regrid water vapor (path)
-O <str>	regrid truth water vapor (path)
-r	set remote processing
-W	regrid water vapor based on altitude [default: pressure grid]

## Example Run:

```
bin/avmp -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.6.nc -o Data/global/EDR/avmp.retr.nc -I  
Data/global/scene/truth.scene.nc -O Data/global/EDR/avmp.truth.nc
```

## Inputs:

Input	Description
retr.mw1.scene.6.nc	Primary input file (15km retrieval from core module)
truth.scene.nc	Truth input profile
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar33 = 33 ;  
nPar20 = 20 ;
```

variables:

```
float Moisture(nSamples, nPar33) ;  
float Latitude(nSamples) ;
```

```

float Longitude(nSamples) ;
char Id(nSamples, nPar20) ;

// global attributes:
:User = "xdong@twister.aer.com" ;
:CreationDate = "Mon Feb 5 09:57:16 2001" ;
:Pressures = 1100.f, 1050.f, 1000.f, 950.f, 900.0001f, 850.0001f,
800.0001f, 750.0001f, 699.9999f, 650.0001f, 600.0001f, 575.0001f, 550.0001f,
524.9999f, 500.0001f, 475.f, 450.f, 425.f, 400.f, 375.f, 350.f, 325.0001f, 300.f, 275.f,
250.f, 225.f, 200.f, 175.f, 150.f, 125.f, 100.f, 75.00001f, 50.f ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nSamples: total number of output profile.  
nPar33: levels of given pressure grid for reporting atmospheric vertical moisture.  
nPar20: length of a ID string.

#### **Variables:**

Moisture: moisture profiles in g/kg, given on pressure grid described in global attributes “Pressures”.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: profile identification string.

### **Source Code Location:**

EDR/avmp

### **Executable Binary Code Location:**

bin/

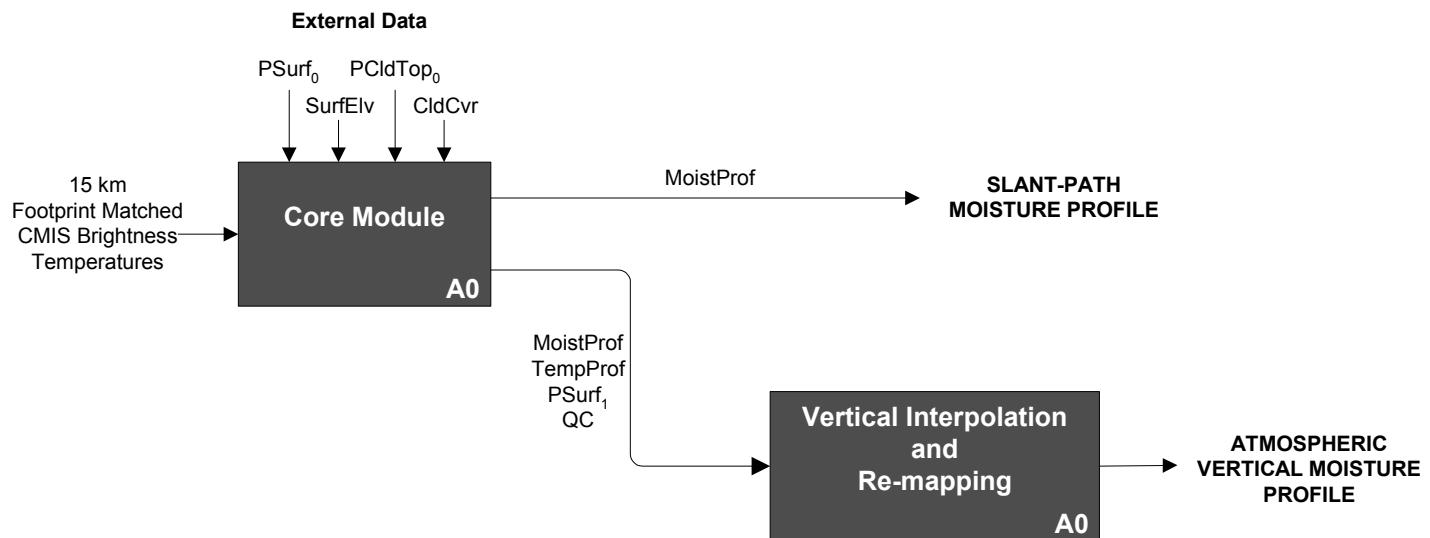
### **Source Codes:**

GNUmakefile  
avmp.C

lib/liburcom.a (source code located in: lib/common)

lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)  
netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



### KEY

#### *Products from CMIS Data*

Input	Description	Source
MoistProf	Atmospheric Moisture Profile	Core Module
TempProf	Atmospheric Temperature Profile	Core Module
$PSurf_1$	Surface Pressure	Core Module

#### *Flags from CMIS Algorithm Output*

Input	Description	Source
QC	Output Production Mode	Core Module

#### *Data from Sources External to CMIS*

Input	Description	Example Source
$PSurf_0$	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElev	Surface Elevation	DTED
$PCldTop_0$	Cloud Top Pressure	VIIRS Cloud Top Pressure EDR
$CldCvr$	Cloud Cover Fraction	VIIRS Cloud Cover EDR

# Atmospheric Vertical Temperature Profile

## Executable Program:

avtp

## Description :

avtp generates atmospheric vertical temperature profiles at given pressure/altitude grids from core retrieval products

## Syntax:

**avtp [-a -C -h -i -o -I -O -r -T]**

## Options:

-a	append data to output file
-C	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name
-o <str>	regrid temperature (path) [default:pressure]
-I <str>	input truth scene file name
-O <str>	regrid truth temperature (path) [default:pressure]
-r	set remote processing
-T	regrid temperature based on altitude

## Example Run:

```
bin/avtp -C etc/ur.conf -i Dara/global/EDR/retr.mw1.scene.2.nc -o  
Data/global/EDR/avtp.retr.nc -I Data/global/scene/truth.scene.nc -O  
Data/global/EDR/avtp.truth.nc
```

## Inputs:

Input	Description
retr.mw1.scene.2.nc	Primary input file ( 40km retrieval from core module)
truth.scene.nc	Truth scene input file
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar37 = 37 ;  
nPar20 = 20 ;
```

variables:

```

float Temperature(nSamples, nPar37) ;
float Latitude(nSamples) ;
float Longitude(nSamples) ;
char Id(nSamples, nPar20) ;

// global attributes:
:User = "xdong@twister.aer.com" ;
:CreationDate = "Mon Feb 5 10:18:30 2001" ;
:Pressures = 1100.f, 1050.f, 1000.f, 950.f, 900.0001f, 850.0001f,
800.0001f, 775.f, 750.0001f, 725.f, 699.9999f, 675.0001f, 650.0001f, 625.0001f,
600.0001f, 575.0001f, 550.0001f, 524.9999f, 500.0001f, 475.f, 450.f, 425.f, 400.f, 375.f,
350.f, 325.0001f, 300.f, 275.f, 250.f, 225.f, 200.f, 175.f, 150.f, 125.f, 100.f, 75.00001f,
50.f;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nSamples: total number of output profile.

nPar37: levels of output pressure grid for calculating atmospheric vertical temperature profile.

nPar20: length for a ID string.

#### **Variables:**

Temperature: atmospheric vertical temperature profiles in Kelvin, given on pressure grid described in global attributes “Pressures”.

Latitude: latitude values in degree north.

Longitude: longitude values in degree east.

Id: profile identification string.

#### **Source Code Location:**

EDR/avtp

#### **Executable Binary Code Location:**

bin/

#### **Source Codes:**

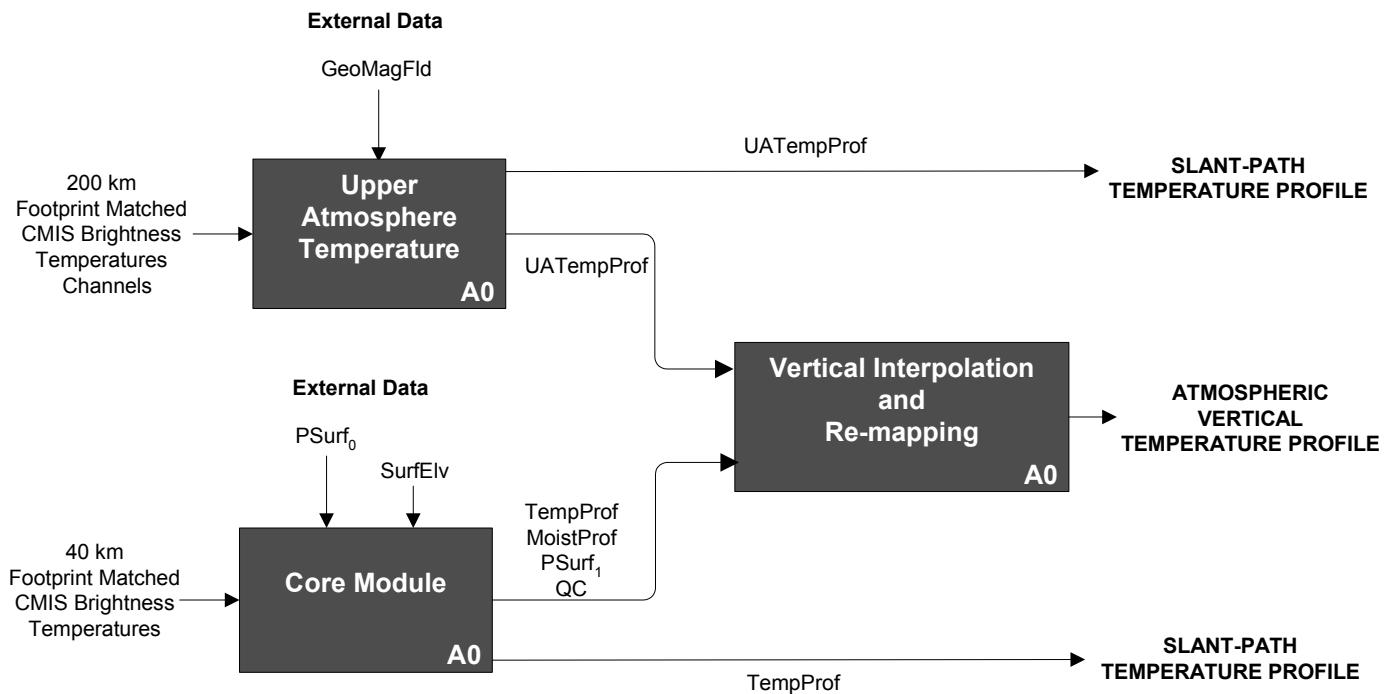
GNUmakefile  
avtp.C

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)

lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
TempProf	Atmospheric Temperature Profile	Core Module
MoistProf	Atmospheric Moisture Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
UATempProf	Upper Atmosphere Temperature Profile	Upper Atmosphere Algorithm
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElev	Surface Elevation	DTED
GeoMagFld	Geomagnetic Field Model	IGRF or DoD

# Cloud Base Height

## Executable Program:

cldBaseH

## Description :

cldBaseH generates cloud base height from core retrieval products

## Syntax:

**cldBaseH [-D -C -h -i -o -r]**

## Options:

-D	turn Fortran debug on
-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name(default:(null))
-o <str>	output scene file name (default:(null))
-r	remote shell option

## Example Run:

```
bin/cldBaseH -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.4.nc -o  
Data/global/EDR/cbh.retr.nc
```

## Inputs:

Input	Description
ur.conf	Configuration file that contains all the default values.
retr.mw1.scene.4.nc	Primary input file (25 km retrieval from core module)

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar20 = 20 ;
```

variables:

```
float CloudBaseHeight(nSamples) ;  
float Latitude(nSamples) ;  
float Longitude(nSamples) ;  
char Id(nSamples, nPar20) ;
```

// global attributes:

```
:User = "xdong@twister.aer.com" ;  
:CreationDate = "Mon Feb 5 11:51:10 2001" ;
```

## **netCDF Descriptions:**

### **Dimensions:**

nSamples: total number of output profile.  
nPar20: length for a ID string.

### **Variables:**

CloudBaseHeight: cloud base height values in kilometer.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: NOAA-88 or other text identification string.

### **Source Code Location:**

EDR/cldBaseH

### **Executable Binary Code Location:**

bin/

### **Source Codes:**

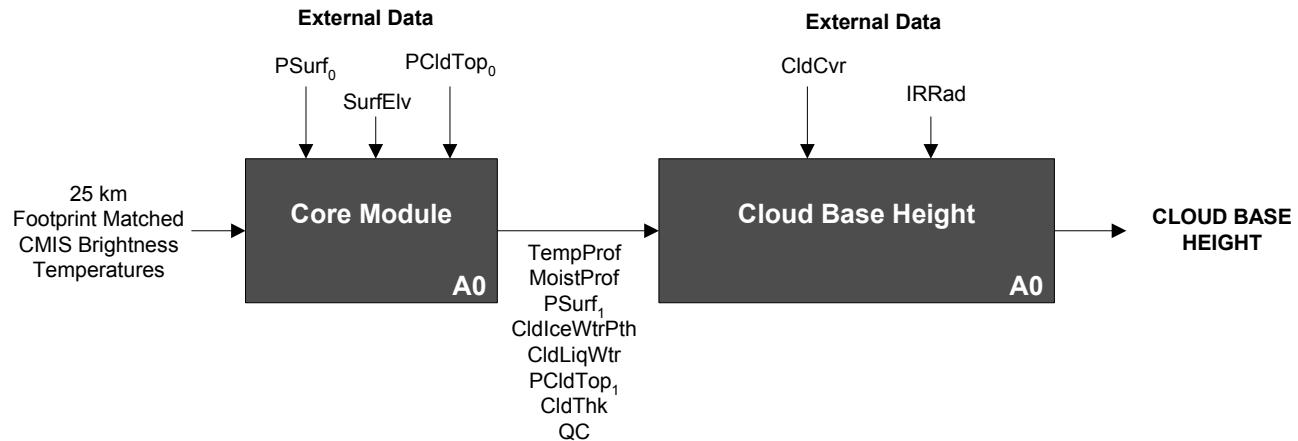
GNUmakefile  
cCldBaseH.C  
fCldBaseH.f

EDR/common/edr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
TempProf	Atmospheric Temperature Profile	Core Module
MoistProf	Atmospheric Moisture Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
CldIceWtrPth	Cloud Ice Water Path	Core Module
CldLiqWtr	Cloud Liquid Water	Core Module
CldThk	Cloud Thickness	Core Module
PCldTop <sub>1</sub>	Cloud Top Pressure	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
CldCvr	Cloud Cover Fraction	VIIRS Cloud Cover EDR
PCldTop <sub>0</sub>	Cloud Top Pressure	VIIRS Cloud Top Pressure EDR
IRRad	IR Radiances	VIIRS

# Cloud Ice Water Path

## Executable Program:

*cmis* for the non-precipitating clouds and *nn4iwp* for precipitating clouds.

## Description :

The cloud ice water path is retrieved in two different regimes. For cirrus cloud case (non-precipitating clouds), we use a physical algorithm, similar to the core module. The state geophysical vector is in this case extended to include the ice water path, the median diameter of the particle size distribution, as well as the cirrus cloud top and the cirrus cloud thickness. The derivatives for the retrieval process are computed using finite differences. The second regime is for precipitating clouds. In this case, the algorithm is based on Neural Networks. The algorithm is applied on the brightness temperatures (read from two files: training and validation sets) and the output is compared to the true values. It is coded in MATLAB. The output is a scatterplot of this comparison.

## Example Run:

1<sup>st</sup> regime (non-precipitating): bin/cmis –run=retr –nprof=100  
2<sup>nd</sup> regime (precipitating) : matlab>nn4iwp

## Inputs:

See Science Code Documentation for CMIS (Volume 1).

## Primary Output File Format :

### 1<sup>st</sup> regime (non-precipitating):

The output is in ASCII format (run/test/retr\_results.asc). For each profile, there is a header containing information about the lat/lon fo the profile, the dat/timem, the number of iteration needed for convergence. It contains 5 columns. For each retrieved profile, it has folling format:

(index, pressure, first guess, retrieval, true value).

The index corresponding to the following:

- 1-40 -----→ temperature Profile
- 41-61 -----→ water vapor profile
- 62 -----→ Tskin
- 63 -----→ surface pressure
- 64-65 ----→ reserved ones
- 66 -----→ cloud top pressure
- 67 -----→ cloud thickness
- 68 -----→ cloud liquid water amount
- 69 -----→ CIWP**
- 70 -----→ DME

71 -----→ variance of particle size distribution (fixed, not retrieved)

72 -----→ cirrus cloud top

73 -----→ cirrus cloud thickness

74-99 -----→ 26 emissivities

2<sup>nd</sup> Regime:

The output is a set of two eps files ( fig1.eps and fig2.eps). They corresponds to scattering profiles representing the comparison of true vs retrieved IWP (training and testing sets respectively).

### **Source Code Location:**

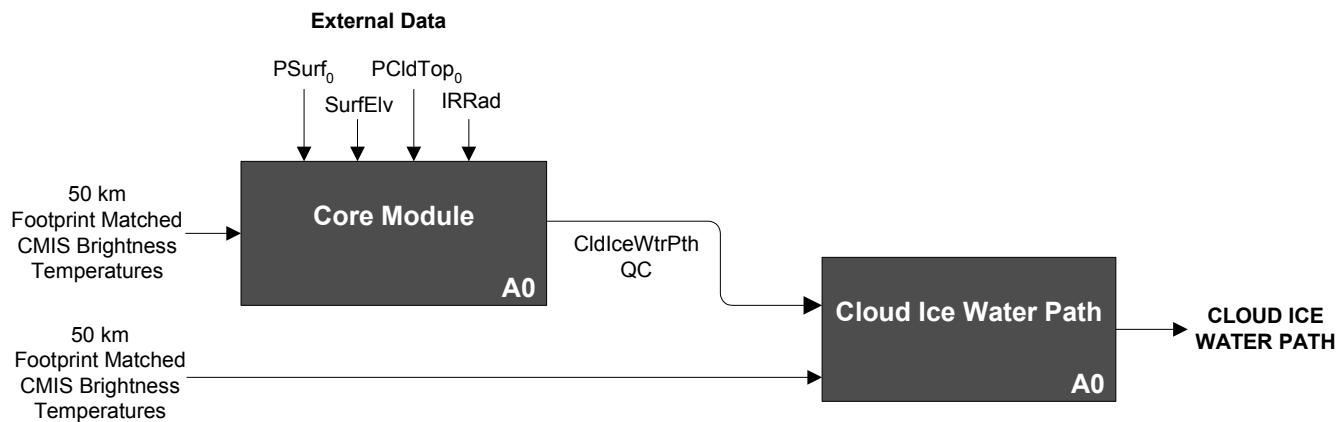
IWP\_nonPrecip\_Xscattering/  
IWP\_Precip\_NeuralNetwork/

See README on above directories for more detail information.

### **Executable Binary Code Location:**

bin/

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
CldIceWtrPth	Cloud Ice Water Path	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
IRRad	IR Radiances	VIIIRS
CldCvr	Cloud Cover Fraction	VIIIRS Cloud Cover EDR
PCldTop <sub>0</sub>	Cloud Top Pressure	VIIIRS Cloud Top Pressure EDR

# Cloud Liquid Water

## Executable Program:

clw

## Description :

clw produce cloud liquid water from core retrieval products

## Syntax:

**clw [-D -C -h -i -o -r]**

## Options:

-D	turn Fortran debug on
-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name(default:(null))
-o <str>	output scene file name (default:(null))
-r	remote shell option

## Example Run:

```
bin/clw -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.5.nc -o  
Data/global/EDR/clw.retr.nc
```

## Inputs:

Input	Description
retr.mw1.scene.5.nc	Primary input file (20km retrieval from core module)
ur.conf	Configuration file that contains the default values.

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar20 = 20 ;
```

variables:

```
float CLW(nSamples) ;  
float Latitude(nSamples) ;  
float Longitude(nSamples) ;  
char Id(nSamples, nPar20) ;
```

// global attributes:

```
:User = "xdong@twister.aer.com" ;  
:CreationDate = "Mon Feb 5 11:58:21 2001" ;
```

**netCDF Descriptions:****Dimensions:**

nSamples: total number of output profile.  
nPar20: length for a ID string.

**Variables:**

CLW: cloud liquid water values in kg/m<sup>2</sup>.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: profile identification string.

**Source Code Location:**

EDR/clw

Executable Binary Code Location:

bin/

**Source Codes:**

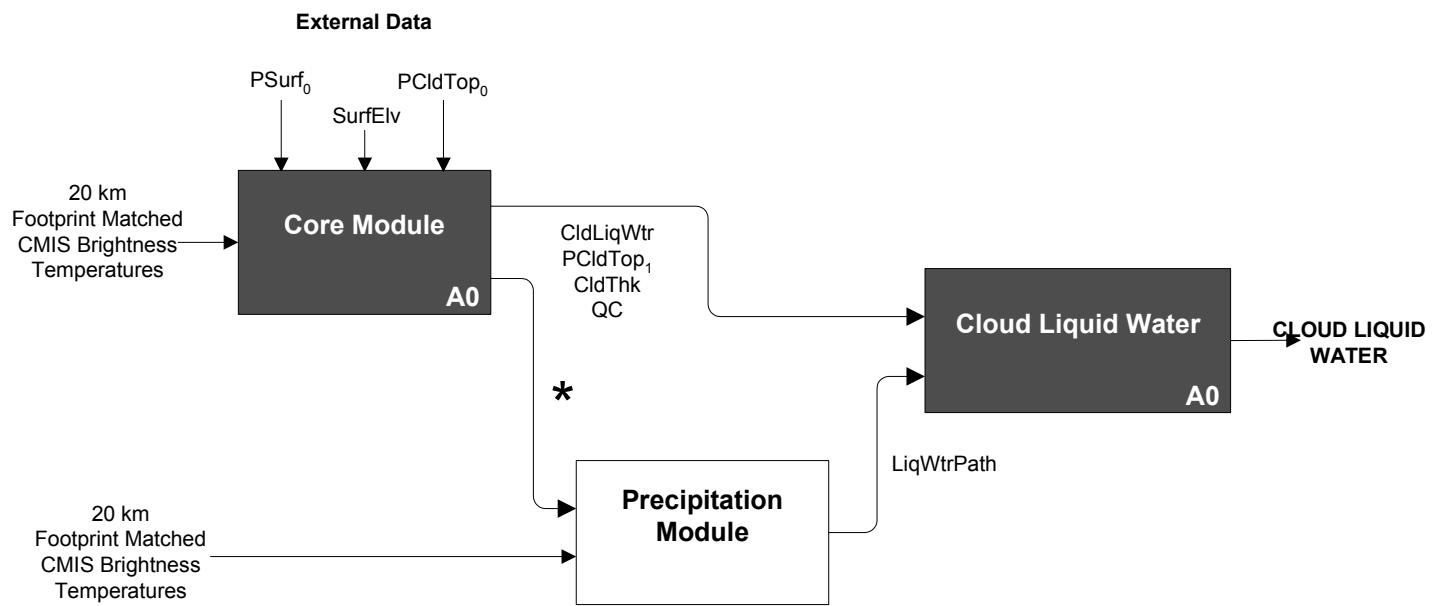
GNUmakefile  
cClw.C  
fClw.f

EDR/common/edr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
CldLiqWtr	Cloud Liquid Water	Core Module
PCldTop <sub>1</sub>	Cloud Top Pressure	Core Module
CldThk	Cloud Thickness	Core Module
LiqWtrPath	Liquid Water Path	Precipitation Module
*	See Precipitation EDR	N/A
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
PCldTop <sub>0</sub>	Cloud Top Pressure	VIIRS Cloud Top Pressure EDR

# Ice Surface Temperature

## Executable Program:

ist

## Description :

ist produces ice surface temperature from sea ice age/concentration products or fresh water age/concentration and core retrieval products

## Syntax:

ist [-C -i -I -m -h -o -r]

## Options:

-C <str>	configuration file (default: etc/ur.conf)
-i <str>	netcdf input path(ice fraction) (default: none)
-I <str>	netcdf input(emissivity) path (default: none)
-m <str>	gridding mapping option (default: map.asc)
-h	help
-o <str>	netcdf output path (default:outIst.nc)
-r	remote shell option

## Example Run:

```
bin/ist -C etc/ur.conf -I Data/global/EDR/retr.mw1.scene.4.nc -i  
Data/global/EDR/ity.retr.nc -o Data/global/EDR/ist.retr.nc
```

## Inputs:

Input	Description
retr.mw1.scene.4.nc	Primary input file (25km retrieval from core module)
ity.retr.nc	First year ice, multi year ice and water fraction from Sea Ice Age/Conc
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nObs = UNLIMITED ; // (20 currently)  
nId = 11 ;  
nTime = 6 ;
```

variables:

```
float iceSurfTemp(nObs) ;  
    iceSurfTemp:long_name = "Ice Surface Temperature" ;  
float Fy(nObs) ;  
    Fy:long_name = "first year ice fraction" ;  
float Water(nObs) ;
```

```

        Water:long_name = "water fraction" ;
char id(nObs, nId) ;
        id:long_name = "Identification Strings" ;
float date(nObs, nTime) ;
        date:long_name = "date and time" ;
float lat(nObs) ;
        lat:long_name = "latitude values" ;
float lon(nObs) ;
        lon:long_name = "longitude values" ;

// global attributes:
:version = "AER 0001" ;
:CreationDate = "Fri Feb 2 11:59:45 2001" ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nObs: total number of output profile.  
nId: length of ID string.  
nTime: length of date variable.

#### **Variables**

iceSurfTemp: ice surface temperature values in Kelvin.  
Fy: first year ice fraction, values are between 0 and 1.  
Water: multi year ice fraction, values are between 0 and 1.  
id: profile identification string.  
date: date and time in year, month, day, hour, minute, second  
format.  
lat: latitude values in degree north.  
Lon: longitude values in degree east.

#### **Source Code Location:**

EDR/ist

#### **Executable Binary Code Location:**

bin/

#### **Source Codes:**

GNUmakefile  
UrIstTemp.C  
NasaTeamObject.C  
NasaTeamObject.H  
IstTemperature.C

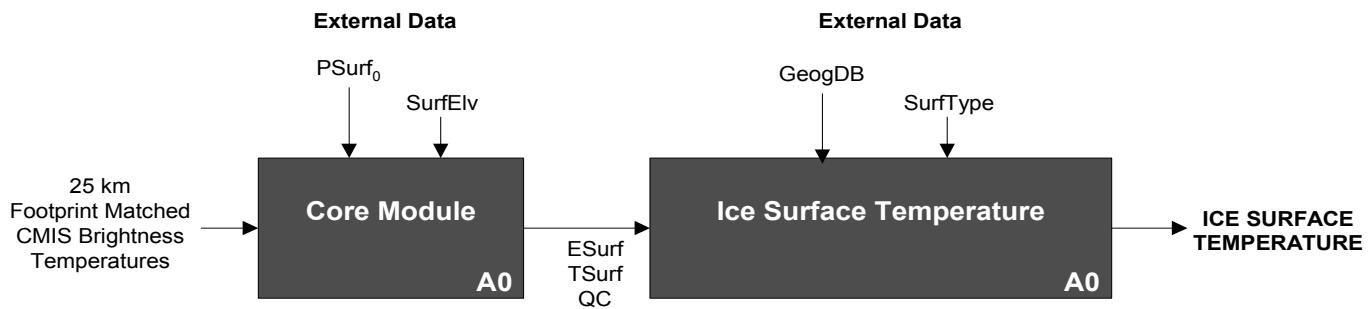
IstTemperature.H

EDR/common/SecondaryEdr.C  
EDR/common/SecondaryEdr.H

lib/liburcom.a (source code located in: lib/common)  
lib/libscene.a (source code located in: lib/scene)  
lib/liburm.a (source code located in: lib/mat)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



### KEY

#### *Products from CMIS Data*

Input	Description	Source
ESurf	Surface Emissivity	Core Module
TSurf	Surface Temperature	Core Module

#### *Flags from CMIS Algorithm Output*

Input	Description	Source
QC	Output Production Mode	Core Module

#### *Data from Sources External to CMIS*

Input	Description	Example Source
SurfType	Surface Type Database	USGS EDC Global Land Cover Characteristics, Version II
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
GeogDB	Geography Database	DTED

# Land Surface Temperature

## Executable Program:

lst

## Description :

lst breeds land surface temperature from core retrieval products

## Syntax:

lst [-D -C -h -i -l -n -o -r]

## Options:

-D	turn Fortran debug on
-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name(default:(null))
-l	enable land only filter based on % Land flag
-n <int>	# of input profiles(default:all)
-o <str>	output scene file name (default:(null))
-r	remote shell option

## Example Run:

```
bin/lst -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.1.nc -o  
Data/global/EDR/lst.retr.nc
```

## Inputs:

Input	Description
ur.conf	Configuration file that contains the default values
retr.mw1.scene.1.nc	Primary input file (50km retrieval from core module)

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar20 = 20 ;
```

variables:

```
float LST(nSamples) ;  
float Latitude(nSamples) ;  
float Longitude(nSamples) ;  
char Id(nSamples, nPar20) ;
```

```
// global attributes:  
:User = "xdong@twister.aer.com" ;  
:CreationDate = "Mon Feb 5 12:06:37 2001" ;
```

### **netCDF Descriptions:**

#### **Dimensions:**

nSamples: total number of output profile.  
nPar20: length of a ID string.

#### **Variables:**

LST: land surface temperature values in Kelvin.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: NOAA-88 or other text identification string.

### **Source Code Location:**

EDR/lst

Executable Binary Code Location:

bin/

### **Source Codes:**

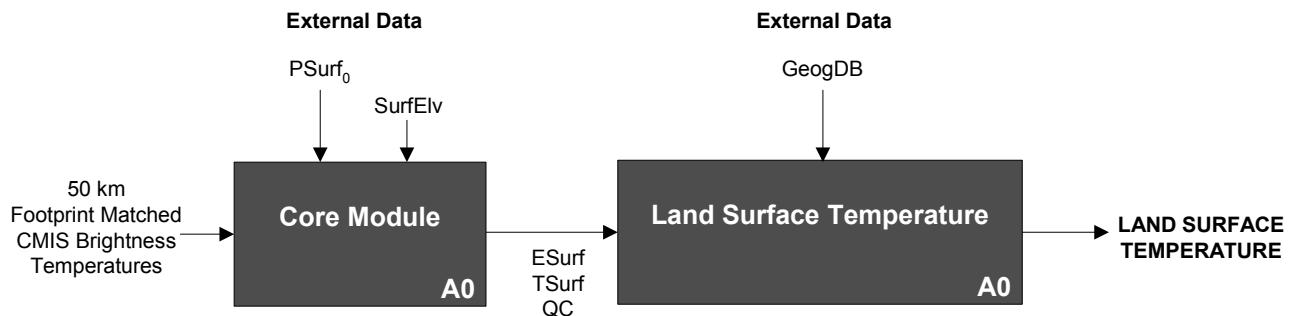
GNUmakefile  
cLst.C  
fLst.f

EDR/common/edr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
ESurf	Surface Emissivity	Core Module
TSurf	Surface Temperature	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>₀</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
GeogDB	Geography Database	DTED

# Pressure Profile

## Executable Program:

pres

## Description :

pres engenders pressure profile according to required altitude from core retrieval products

## Syntax:

**pres [-C -h -i -I -o -O -r -w]**

### Options:

-C	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name
-I <str>	input truth file name
-o <str>	output scene file name (default:pressure.nc)
-O <str>	output truth file name (default: none)
-r	set remote processing
-w	append data to output file

## Example Run:

```
bin/pres -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.4.nc -o  
Data/global/EDR/pres.retr.nc -I Data/global/scene/truth.scene.nc -O  
Data/global/EDR/pres.truth.nc
```

## Inputs:

Input	Description
retr.mw1.scene.4.nc	Primary input file (25km retrieval from core module)
truth.scene.nc	Truth input profile
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar31 = 31 ;  
nPar20 = 20 ;
```

variables:

```
float Pressure(nSamples, nPar31) ;  
float Latitude(nSamples) ;  
float Longitude(nSamples) ;
```

```

char Id(nSamples, nPar20) ;

// global attributes:
:User = "xdong@twister.aer.com" ;
:CreationDate = "Mon Feb 5 10:47:15 2001" ;
:Altitude = 30.f, 29.f, 28.f, 27.f, 26.f, 25.f, 24.f, 23.f, 22.f, 21.f, 20.f, 19.f,
18.f, 17.f, 16.f, 15.f, 14.f, 13.f, 12.f, 11.f, 10.f, 9.f, 8.f, 7.f, 6.f, 5.f, 4.f, 3.f, 2.f, 1.f,
0.f ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nSamples: total number of output profile.  
nPar31: length of each pressure variable.  
nPar20: length of a ID string.

#### **Variables:**

Pressure: pressure profiles in millibar, given on altitude grid in kilometer specified at global attribute “Altitude”.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: profile identification string.

#### **Source Code Location:**

EDR/pressure

#### **Executable Binary Code Location:**

bin/

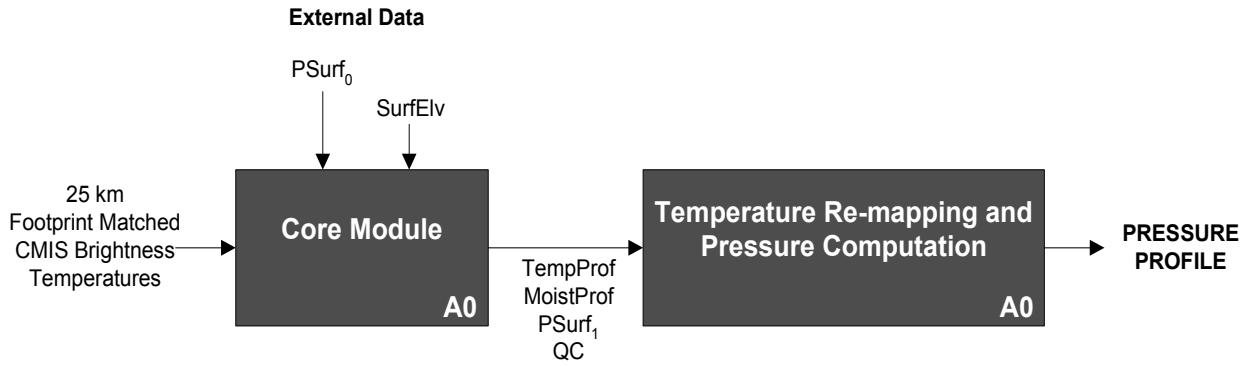
#### **Source Codes:**

GNUmakefile  
pressure.C

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

#### **Function Flow Diagram:**



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
TempProf	Atmospheric Temperature Profile	Core Module
MoistProf	Atmospheric Moisture Profile	Core Module
PSurf₁	Surface Pressure	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf₀	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED

# Precipitable Water

## Executable Program:

precipw

## Description :

precipw yields precipitable water from core retrieval products

## Syntax:

**precipw [-C -h -i -o -n -r]**

## Options:

-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name(default:(null))
-o <str>	output scene file name (default:(null))
-n <int>	# of input profiles(default:all)
-r	remote shell option

## Example Run:

```
bin/precipw -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.4.nc -o  
Data/global/EDR/pw.retr.nc
```

## Inputs:

Input	Description
retr.mw1.scene.4.nc	Primary input file (25km retrieval from core module)
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar20 = 20 ;
```

variables:

```
float PRECIPW(nSamples) ;  
float Latitude(nSamples) ;  
float Longitude(nSamples) ;  
char Id(nSamples, nPar20) ;
```

// global attributes:

```
:User = "xdong@twister.aer.com" ;  
:CreationDate = "Mon Feb 5 10:59:32 2001" ;
```

## **netCDF Descriptions:**

### **Dimensions:**

nSamples: total number of output profile.  
nPar20: length of a Id string.

### **Variables:**

PRECIPW: precipitable water values in kg/m<sup>2</sup>.  
Latitude: latitude values in degree north.  
Longitude: longitude values in degree east.  
Id: profile identification string.

### **Source Code Location:**

EDR/precipWater

### **Executable Binary Code Location:**

bin/

### **Source Codes:**

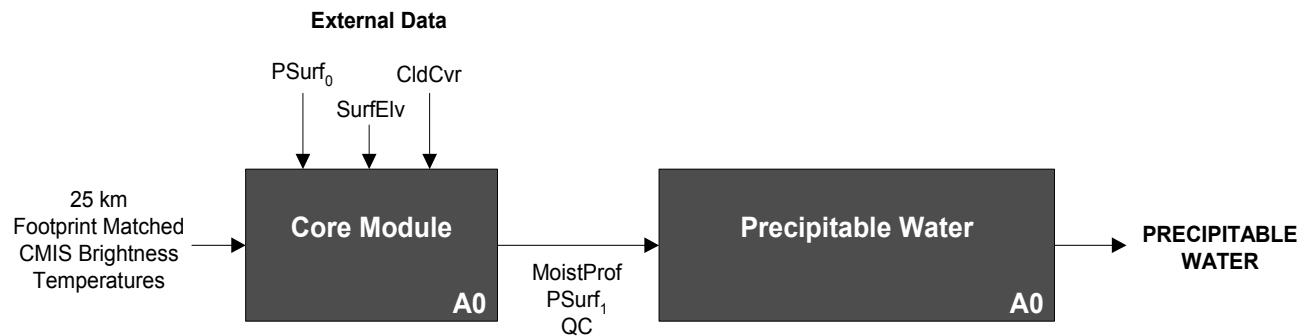
GNUmakefile  
cPrecipw.C  
fPrecipw.f

EDR/common/edr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
MoistProf	Atmospheric Moisture Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElev	Surface Elevation	DTED
CldCvr	Cloud Cover Fraction	VIIIRS Cloud Cover EDR

# Sea Ice Age/Fresh Water Ice Age

## Executable Program:

ity

### Description :

ity generates sea ice age and/or fresh water ice age from core retrieval products

### Syntax:

ity [-p -C -i -t -f -r -h -o]

### Options:

-p <str>	parameter path (default: coeff.conf)
-C <str>	configuration file (default: etc/ur.conf)
-i <str>	netcdf input path (default: none)
-t <str>	Emissivity Flag(0)/Brightness Temperature Flag(1) (default: 0)
-f <str>	Sea ice (0)/Fresh Water Flag(1) (default:0)
-r	remote shell option
-h	help
-o <str>	netcdf output path (default:outType.nc)

### Example Run:

```
Bin/ity -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.5.nc -o  
Data/global/EDR/ity.retr.nc
```

### Inputs:

Input	Description
Retr.mw1.scene.5.nc	Primary input file (20km retrieval from core module)
ur.conf	Configuration file that contains the default values
Coefficiency	Constant Vectors

### Primary Output File Format (netCDF CDL description):

dimensions:

```
nObs = UNLIMITED ; // (20 currently)  
nId = 11 ;  
nTime = 6 ;
```

variables:

```
float Fy(nObs) ;  
    Fy:long_name = "first year ice fraction" ;  
float My(nObs) ;  
    My:long_name = "multi year ice fraction" ;
```

```

float water(nObs) ;
    water:long_name = "water fraction" ;
float Ice(nObs) ;
    Ice:long_name = "total ice fraction" ;
int IceType(nObs) ;
    IceType:long_name = "ice type, 0-ocean, 1-Fy, 2-My" ;
char id(nObs, nId) ;
    id:long_name = "Identification Strings" ;
float date(nObs, nTime) ;
    date:long_name = "date and time" ;
float lat(nObs) ;
    lat:long_name = "latitude values" ;
float lon(nObs) ;
    lon:long_name = "longitude values" ;

// global attributes:
:version = "AER 0001" ;
:CreationDate = "Fri Feb 2 11:59:45 2001" ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nObs: total number of output profile.  
nId: length of a ID string.  
nTime: length of a date variable.

#### **Variables:**

Fy: first year ice fraction, values are between 0 and 1.  
My: multi year ice fraction, values are between 0 and 1.  
Water: water fraction, values are between 0 and 1.  
Ice: total ice fraction (first year ice plus multi year ice),  
values are between 0 and 1.  
IceType: dominant type. 0 for ocean, 1 for first year ice  
and 2 for multi year ice.  
id: NOAA-88 or other text identification string.  
date: date and time in year, month, day, hour, minute, second  
format.  
lat: latitude values in degree north.  
lon: longitude values in degree east.

#### **Source Code Location:**

EDR/ist

**Executable Binary Code Location:**

bin/

**Source Codes:**

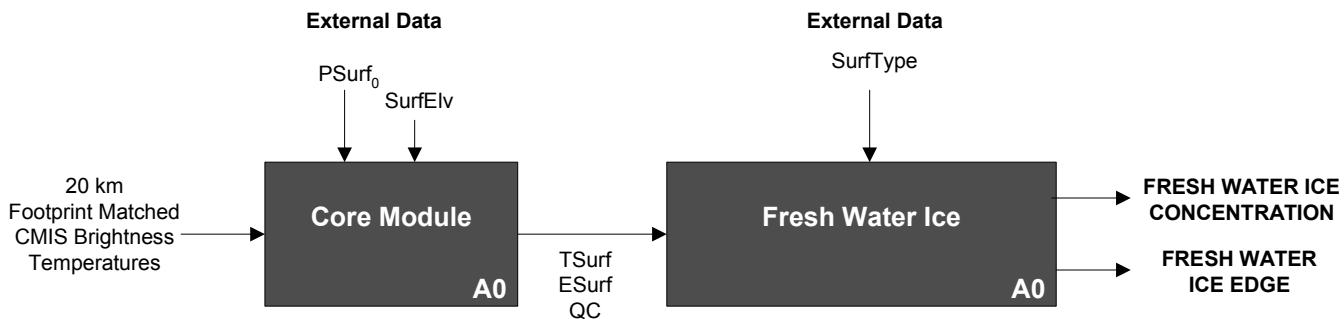
GNUmakefile  
SeaIceAge.C  
NasaTeamObject.C  
NasaTeamObject.H

EDR/common/SecondaryEdr.C  
EDR/common/SecondaryEdr.H

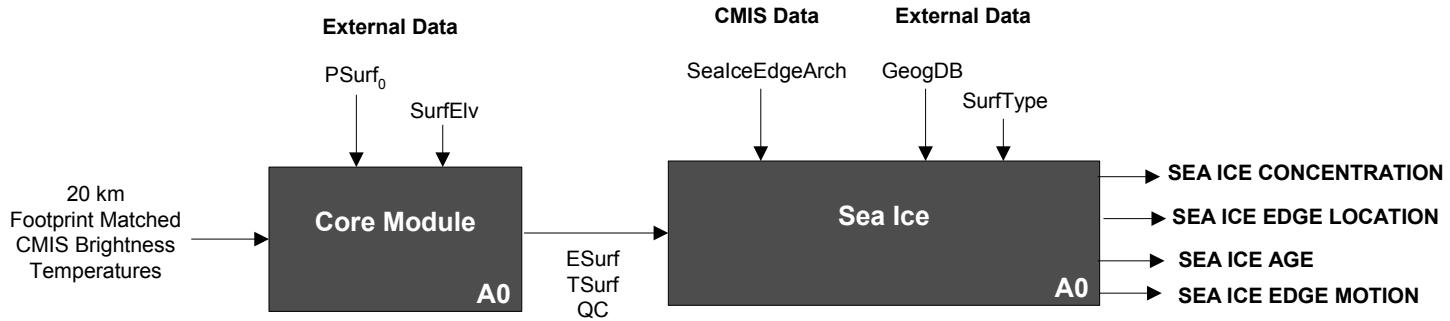
lib/liburcom.a (source code located in: lib/common)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagrams:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
TSurf	Surface Temperature	Core Module
ESurf	Surface Emissivity	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
SurfType	Surface Type Database	USGS EDC Global Land Cover Characteristics, Version II
PSurf <sub>₀</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
Tsurf	Surface Temperature	Core Module
ESurf	Surface Emissivity	Core Module
SealceEdgeArch	Archived Sea Ice Edge	Core Module/Sea Ice Algorithm
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>₀</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED
SurfType	Surface Type Database	USGS EDC Global Land Cover Characteristics, Version II

# Soil Moisture

## Executable Program:

smt

## Description :

smt engenders soil moisture from core retrieval products

## Syntax:

**smt [-C -h -e -i -o -p -r -s -t]**

## Options:

-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-e <str>	Coefficiency path(default:nedt_9904.dat)
-i <str>	netcdf input file(default:none)
-o <str>	NETCDF output file path(default:output.nc)
-n <int>	Number of Input profiles
-p <str>	cloud atmospheric file path(default:cldatm.out)
-r	remote shell option
-s <str>	standard atmospheric file path(default:stdatm.out)
-t <str>	control file path(default:soiltop.conf)

## Example Run:

```
bin/smt -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.2.nc -o  
Data/global/EDR/sm.retr.nc
```

## Inputs:

Input	Description
Retr.mw1.scene.2.nc	Primary input file (40km retrieval from core module)
Ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nObs = UNLIMITED ; // (20 currently)  
nId = 11 ;  
nTime = 6 ;
```

variables:

```
float SoilMoist(nObs) ;  
SoilMoist:long_name = "soil moisture" ;  
float VegWater(nObs) ;
```

```

VegWater:long_name = "vegation water content of soil" ;
char id(nObs, nId) ;
    id:long_name = "Identification Strings" ;
float date(nObs, nTime) ;
    date:long_name = "date and time" ;
float lat(nObs) ;
    lat:long_name = "latitude values" ;
float lon(nObs) ;
    lon:long_name = "longitude values" ;

// global attributes:
:version = "AER 0001" ;
:CreationDate = "Wed Feb 7 12:14:42 2001" ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nObs: total number of output profile.  
nId: length of a ID string.  
nTime: length of a date variable.

#### **Variables:**

SoilMoist: soil moisture values in percentage (volume of moisture over volume of soil). SoilMoist= -9 when algorithm does not converge after maximum number of iterations (current 200).  
VegWater: vegetation water content values in kg/m<sup>2</sup>. VegWater =-9  
When algorithm does not converge after maximum number of iterations (current 200).  
id: profile identification string.  
date: date and time in year, month, day, hour, minute, second format.  
lat: latitude values in degree north.  
lon: longitude values in degree east.

#### **Source Code Location:**

EDR/smt

#### **Executable Binary Code Location:**

bin/

#### **Source Codes:**

GNUmakefile

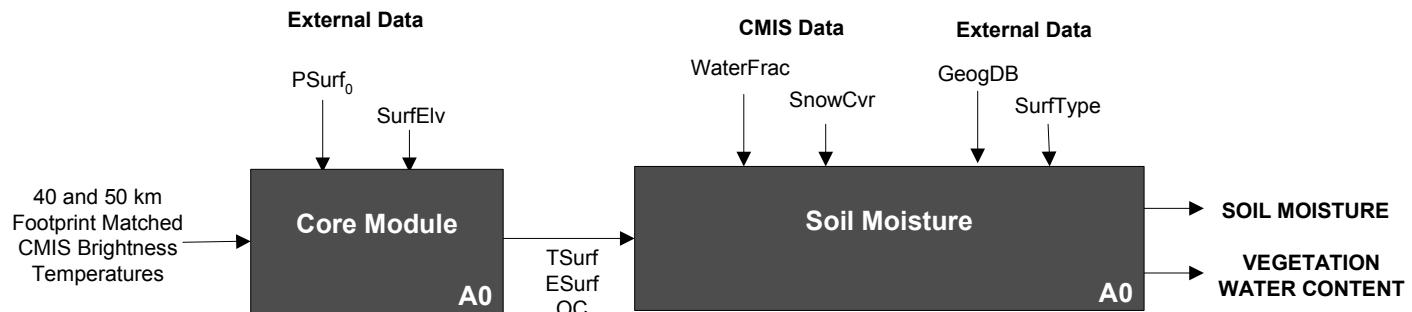
Roughks.C  
Roughks.H  
SMAlgorithm.C  
SMAlgorithm.H  
SMForward.C  
SMForward.H  
SMReader.C  
SMReader.H  
SoilMoisture.C  
SoilTop.C  
SoilTop.H  
VarNedt.C  
VarNedt.H

EDR/common/SecondaryEdr.C  
EDR/common/SecondaryEdr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
ESurf	Surface Emissivity	Core Module
TSurf	Surface Temperature	Core Module
WaterFrac	Open Water Fraction	Core Module/Vegetation/Surface Type Algorithm
SnowCvr	Snow Cover	Core Module/Snow Cover/Depth Algorithm
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
GeogDB	Geography Database	DTED
SurfType	Surface Type Database	USGS EDC Global Land Cover Characteristics, Version II
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED

# Snow Cover/Depth

## Executable Program:

snowcover\_main.m

## Description :

snowcover\_main.m matLab script runs retrieval tests using brightness temperature (provided).

## Syntax:

none

## Example Run:

snowcover\_main

## Inputs:

Input	Description
gcells99035x035.mat	snowcover true value, latitude, longitude and brightness temperatures
snowarch.mat	Previous retrieval archive like gravity ratio and snow cover

## Primary Output File Format:

Matlab Binary-Format File: shat99035x035ir1ipf2in2.mat

## Source Code Location:

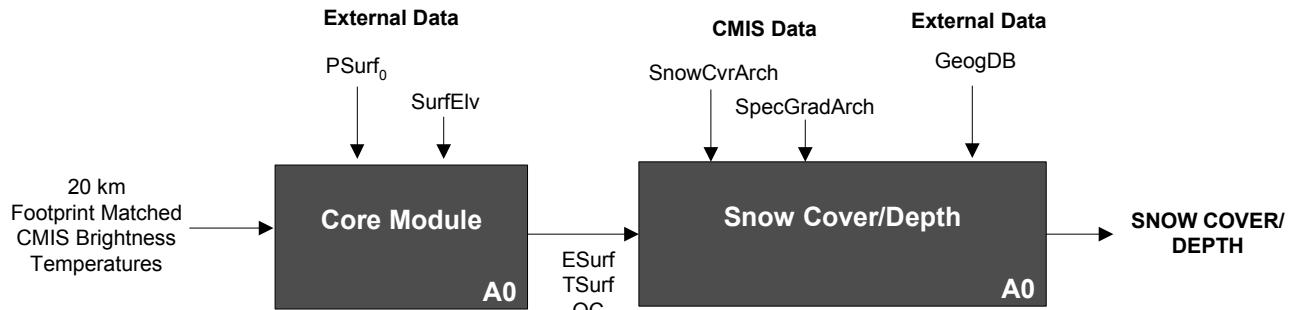
EDR/scover/matlab

## Source Codes:

annotemap.m  
fsnow\_alg.m  
fssmigrid.m  
getdata4map.m  
getspectrals.m  
grodyfunc.m  
mapsnow.m  
plotphys.m  
snowcover\_main.m

wet\_alg.m

## Function Flow Diagram:



### KEY

#### *Products from CMIS Data*

Input	Description	Source
ESurf	Surface Emissivity	Core Module
TSurf	Surface Temperature	Core Module
SnowCvrArch	Archived Snow Cover	Core Module/Snow Cover/Depth Algorithm
SpecGradArch	Archived Spectral Gradients	Core Module/Snow Cover /Depth Algorithm

#### *Flags from CMIS Algorithm Output*

Input	Description	Source
QC	Output Production Mode	Core Module

#### *Data from Sources External to CMIS*

Input	Description	Example Source
GeogDB	Geography Database	DTED
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED

# Total Water Content

## Executable Program:

twc

## Description :

twc generates layered total water content from core retrieval products

## Syntax:

**twc [-a -C -h -i -o -n -r]**

## Options:

-a	append data to output file
-C <str>	configuration file path (default: etc/ur.conf )
-h	help
-i <str>	input scene file name(default: none)
-o <str>	total water content output path(default: twc.nc)
-n <int>	# of input profiles(default:all)
-r	remote shell option

## Example Run:

```
bin/twc -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.5.nc -o  
Data/global/EDR/twc.retr.nc
```

## Inputs:

Input	Description
retr.mw1.scene.5.nc	Primary input file (20km retrieval from core module)
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nSamples = UNLIMITED ; // (20 currently)  
nPar7 = 7 ;  
nPar20 = 20 ;
```

variables:

```
float LayerVaporTWC(nSamples, nPar7) ;  
float LayerLiquidTWC(nSamples, nPar7) ;  
float LayerIceTWC(nSamples, nPar7) ;  
float LayerTWC(nSamples, nPar7) ;  
char Id(nSamples, nPar20) ;
```

// global attributes:

```
:User = "xdong@twister.aer.com" ;
:CreationDate = "Mon Feb 5 09:27:07 2001" ;
:Altitude = 21.f, 18.f, 15.f, 12.f, 9.f, 6.f, 3.f, 0.f ;
```

### **netCDF Descriptions:**

#### **Dimensions:**

nSamples: total number of output profile.  
nPar7: length of the layer liquid water.  
nPar20: length of a ID string.

#### **Variables:**

LayerVaporTWC: vapor water content profiles in kg/m<sup>2</sup>, given  
on altitude grid in kilometer specified in global  
attributes “Altitude”.

LayerLiquidTWC: liquid water content profiles in kg/m<sup>2</sup>, given  
on altitude grid in kilometer specified in global  
attributes “Altitude”.

LayerVaporTWC: ice water content profiles in kg/m<sup>2</sup>, given  
on altitude grid in kilometer specified in global  
attributes “Altitude”.

LayerTWC: total water content profiles in kg/m<sup>2</sup>, given  
on altitude grid in kilometer specified in global  
attributes “Altitude”.

Id: profile identification string.

### **Source Code Location:**

EDR/twc

### **Executable Binary Code Location:**

bin/

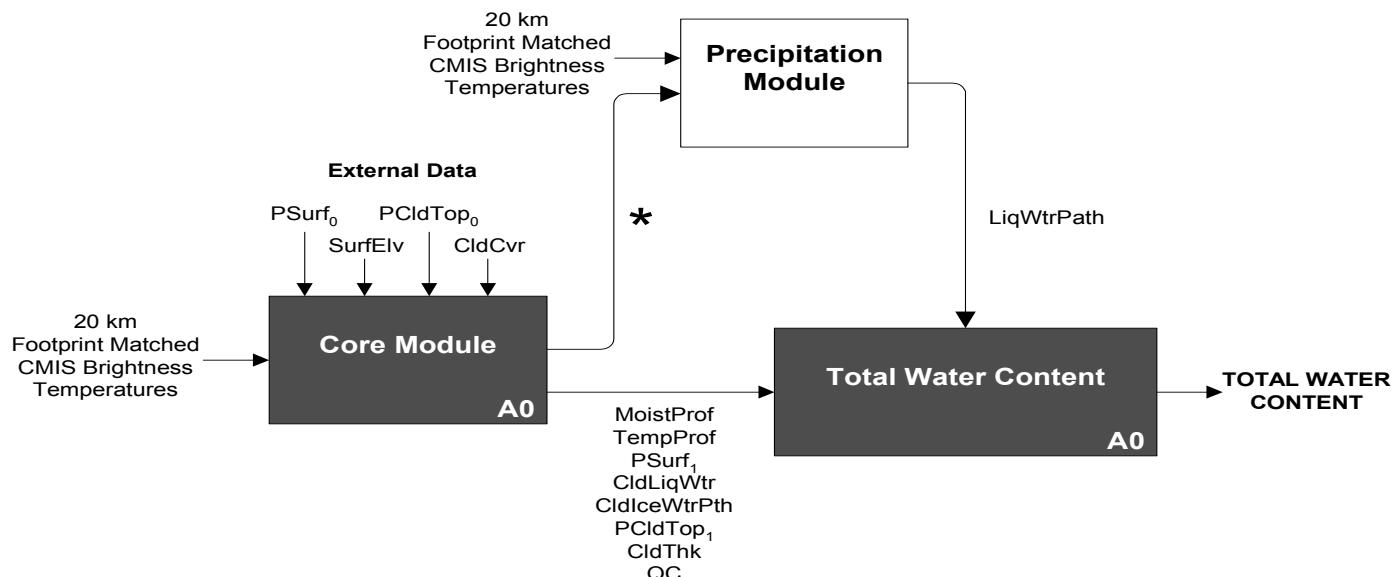
### **Source Codes:**

GNUmakefile  
TotalWaterCont.C

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
MoistProf	Atmospheric Moisture Profile	Core Module
TempProf	Atmospheric Temperature Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
CldLiqWtr	Cloud Liquid Water	Core Module
CldIceWtrPth	Cloud Ice Water Path	Core Module
PCldTop <sub>1</sub>	Cloud Top Pressure	Core Module
CldThk	Cloud Thickness	Core Module
LiqWtrPath	Liquid Water Path	Precipitation Module
*	See Precipitation EDR	N/A
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMO/NOGAPS
SurfElv	Surface Elevation	DTED
CldCvr	Cloud Cover Fraction	VIIRS Cloud Cover EDR
PCldTop <sub>0</sub>	Cloud Top Pressure	VIIRS Cloud Top Pressure EDR

# Vegetation Surface Type

## Executable Program:

vst

## Description :

vst produces vegetation surface type from core retrieval products

## Syntax:

**vst [-C -h -i -l -n -p -o -r]**

## Options:

- C <str> configuration file path (default: etc/ur.conf )
- h help
- i <str> input file path(default: none)
- l filter based on pLand flag
- n <int> # of input profiles(default:all)
- p <str> parameter file path(default: landSurfaceCoeff.nc)
- o <str> output file path(default: VST.nc)
- r remote shell option

## Example Run:

```
bin/vst -C etc/ur.conf -i Data/global/EDR/retr.mw1.scene.5.nc -o  
Data/global/EDR/vst.retr.nc
```

## Inputs:

Input	Description
retr.mw1.scene.5.nc	Primary input file (20km retrieval from core module)
ur.conf	Configuration file that contains the default values

## Primary Output File Format (netCDF CDL description):

dimensions:

```
nObs = UNLIMITED ; // (20 currently)  
nId = 11 ;  
nTime = 6 ;
```

variables:

```
int LandType(nObs) ;  
    LandType:long_name = "Vegetation Surface Type" ;  
char id(nObs, nId) ;
```

```

        id:long_name = "Identification Strings" ;
        float date(nObs, nTime) ;
            date:long_name = "date and time" ;
        float lat(nObs) ;
            lat:long_name = "latitude values" ;
        float lon(nObs) ;
            lon:long_name = "longitude values" ;

// global attributes:
:version = "AER 0001" ;
:CreationDate = "Fri Feb 2 11:59:51 2001" ;

```

### **netCDF Descriptions:**

#### **Dimensions:**

nObs: total number of output profile.  
nId: length of a ID string.  
nTime: length of a date variable.

#### **Variables:**

LandType: vegetation surface type. It is an enumerated number.  
Detail types are:  
Land=0,enForest=1,ebForest=2,dnForest=3,dbForest=4,  
mForest=5,cShrub=6,oShrub=7,oSavannas=8,wSavannas=9,  
Grass=10,Wetlands=11,Crops=12,Urban=13,  
Vegetation=14,SnowIce=15,Barren=16,Water=17.  
id: profile identification string.  
date: date and time in the year, month, day, hour, minute, second  
format.  
lat: latitude values in degree north.  
lon: longitude values in degree east.

#### **Source Code Location:**

EDR/vst

#### **Executable Binary Code Location:**

bin/

#### **Source Codes:**

GNUmakefile  
LandSurfWriter.C  
LandSurfWriter.H  
VegeSurfType.C

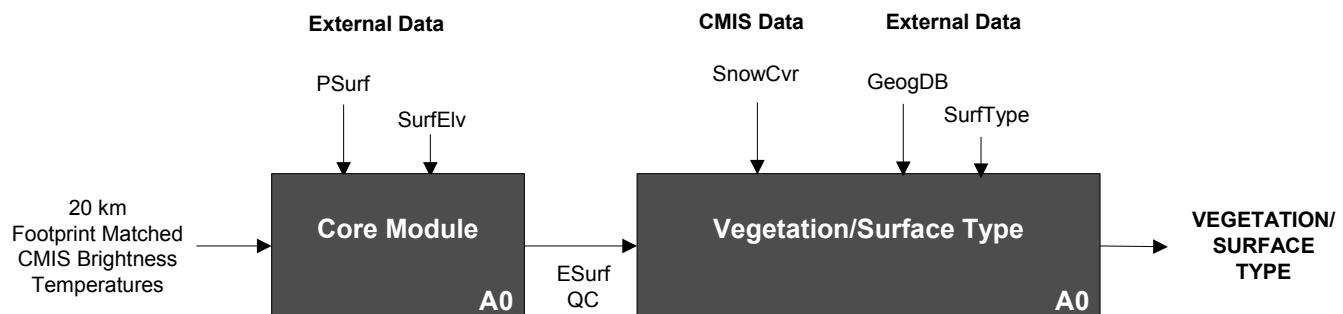
VegetationSurf.C  
VegetationSurf.H

EDR/common/SecondaryEdr.C  
EDR/common/SecondaryEdr.H

lib/liburcom.a (source code located in: lib/common)  
lib/liburm.a (source code located in: lib/mat)  
lib/libscene.a (source code located in: lib/scene)

netCDF version 3.4 libraries (Unidata)

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
ESurf	Surface Emissivity	Core Module
SnowCvr	Snow Cover	Core Module/Snow Cover/Depth Algorithm
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
GeogDB	Geography Database	DTED
SurfType	Surface Type Database	USGS EDC Global Land Cover Characteristics, Version II
PSurf	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElv	Surface Elevation	DTED

# Wind Speed/Wind Stress/Wind Direction Sea Surface Temperature

**Source Code location:** EDR/windspeed

## GENERAL Code Packaging

The code package consists of 2 parts:

1. **ambig\_rem\_1.f**    *DRIVER ROUTINE+ subroutines for computing statistics*
2. **mfilter.f**            *Median Filter + necessary Subroutines*

### Driver Routine + Dependencies

The driver routine **PROGRAM ambig\_rem\_1**

reads in the gridded ambiguous wind field, calls the MF and provides output statistics as well as output wind fields for diagnostics. It calls the following subroutines:

- fdclosest
- mfilter
- crosstalk\_bin
- crosstalk\_skill

### Median Filter + Dependencies

The major subroutine is **mfilter** (which is called by the driver routine ambig\_rem\_1).

**mfilter** calls the functions

- CF
- Weight

and the sorting subroutines

- shuffle
- indx
- sort

## External Files

FILE NAME	FOLDER	I/O	UNIT
AMBIG_FILEDS_O1.dat	fields	I	7
STATISTICS_MF_O1.dat		O	8
SELECTED_FILEDS_O1.dat	fields	O	9

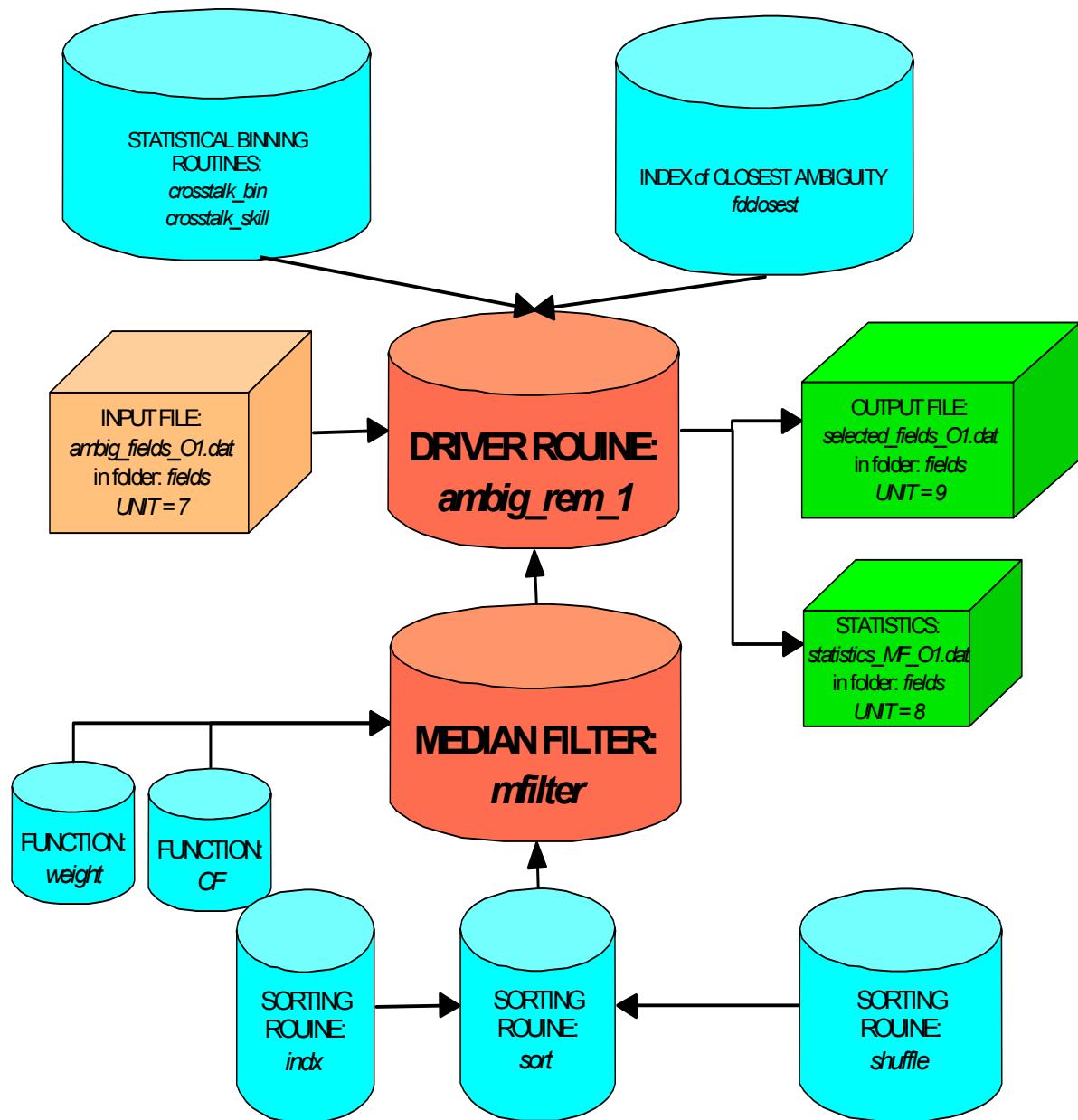


Figure 1: Dependency Structure for Ambiguity Removal Code.

## DRIVER ROUTINE

### PROGRAM:

ambig\_rem\_1

### USAGE:

PROGRAM ambig\_rem\_1

### DESCRIPTION:

(1) Read in ambiguous test wind fields from input file. The test fields are gridded on an  $(nx, ny)$  grid. Each grid point  $(ix, iy)$  contains:

- a. the grid coordinates  $ix$  and  $iy$
- b. the number of ambiguities  $n\_amb(ix, iy)$
- c. the true wind speed  $wind\_t(ix, iy)$
- d. up to 4 ambiguous wind speeds  $WIND\_AMB(1:4, ix, iy)$
- e. the true wind direction (relative to N)  $phi\_t(ix, iy)$
- f. up to 4 ambiguous wind speeds  $PHI\_AMB(1:4, ix, iy)$

The test fields were taken from the ascending part of the NCEP Orbit 1 (for details c.f. CMIS Ocean ATBD by C. Smith, F. Wentz and T. Meissner). The retrievals were done using the RSS wind vector retrieval routines. The fields were put on a rectangular grid with dimensions:  $nx=75$  and  $ny=650$  dim grid, which corresponds to taking every 2<sup>nd</sup> scan and every 4<sup>th</sup> cell in each scan.

If  $n\_amb < 4$ , the remaining entries in  $WIND\_AMB$  and  $PHI\_AMB$  were declared as *missing value* = + 9999.

At invalid grid points (land, ice, rain, bad retrieval) the whole fields  $WIND\_AMB$  and  $PHI\_AMB$  were declared as missing values and  $n\_amb(ix, iy) = 0$ . The fields are read in form the *ASCII File AMBIG\_FIELDS.DAT* in the folder **fields** (unit = 7).

(2) Compute  $u$  (*zonal*) and  $v$  (*meridional*) components for each ambiguity.

(3) Compute index for closest ambiguity at each grid point:  $CLOSEST(ix, iy)$

(4) Pass through MF index for selected field at each grid point:  $SELECTED(ix, iy)$ .

(5) determine **MF SKILL IMPROVEMENT CODE IMPRV** for diagnostics (C = closest, S= selected):

<b>IMPRV</b>	<b>CONDITION</b>	<b>EFFECT on SKILL</b>
0	$C(ix, iy) = S(ix, iy) = 1$	SAME
+1	$C(ix, iy) \neq 1$ and $C(ix, iy) = S(ix, iy)$	improved
-1	$C(ix, iy) \neq 1$ and $C(ix, iy) \neq S(ix, iy)$	SAME, MF failed
-2	$C(ix, iy) = 1$ and $C(ix, iy) \neq S(ix, iy)$	decreased

(6) If  $\text{IMPRV}(ix, iy) \neq 0$  the selected wind field is written to the *ASCII File SELECTED\_FIELDS.DAT* in the folder **fields** (unit = 9).

Each grid point  $(ix, iy)$  contains:

- a. the grid coordinates  $ix$  and  $iy$
- b. the skill improvement flag  $IMPRV$
- c. the number of ambiguities  $n\_amb(ix, iy)$
- d. the index for the closest ambiguity :  $closest(ix, iy)$
- e. the index for the selected ambiguity:  $selected(ix, iy)$ .  
at invalid grid points we set  $selected(ix, iy) = closest(ix, iy) = 1$
- f. the true wind speed  $wind\_t(ix, iy)$
- g. the closest wind speed  $wind\_amb(closest(ix, iy), ix, iy)$
- h. the 1st ranked wind speed  $wind\_amb(1, ix, iy)$
- i. the selected wind speed  $wind\_amb(selected(ix, iy), ix, iy)$
- j. the true wind direction  $phi\_t(ix, iy)$
- k. the closest wind direction  $phi\_amb(closest(ix, iy), ix, iy)$
- l. the 1st ranked wind direction  $phi\_amb(1, ix, iy)$
- m. the selected wind direction  $phi\_amb(selected(ix, iy), ix, iy)$

(7) Compute statistics:

- a. BIAS + SDEV for selected wind speed
- b. BIAS + SDEV for closest, 1st ranked and selected wind direction
- c. SKILL for 1st ranked and SELECTED ambiguity

## MEDIAN FILTER CODE

### SUBROUTINE:

mfilter

### USAGE:

**CALL mfilter (itermax,wsize,nx,ny,N\_AMB,U\_AMB,V\_AMB,  
SELECTED,JITER,XGAIN,IRETURN)**

### DESCRIPTION:

Circular median filter for maximal 4 ambiguities on grid ( $nx, ny$ ).

### INPUT:

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>itermax</i>	Maximum number of iterations allowed	INTEGER(4)	Scalar		> 2
<i>winsize</i>	MF window size	INTEGER(4)	Scalar		Odd Number <i>Winsize</i> $\leq \text{Min}(nx, ny)$
<i>nx</i>	X dimension	INTEGER(4)	Scalar		> 0
<i>ny</i>	Y dimension	INTEGER(4)	Scalar		> 0
<i>n_amb</i>	Number of ambiguities	INTEGER(4)	Array (nx,ny)		[1,4] for invalid grid point (land, ice, rain, bad retrieval) put <i>n_amb</i> ( <i>ix, iy</i> ) = 0
<i>U_AMB</i>	U (zonal) component of ambiguous wind field	REAL(4)	Array (nx,ny)	m/s	] -30, 30 [ <ul style="list-style-type: none"> <li>• for invalid grid point (land, ice, rain, bad retrieval) put: <i>u_amb</i>(<i>l:4, ix, iy</i>) = missing = + 9999</li> <li>• if <i>n_amb</i>(<i>ix, iy</i>) &lt; 4 put <i>u_amb</i>(<i>n_amb+1:4, ix, iy</i>) = missing = + 9999</li> </ul>
<i>V_AMB</i>	V (meridional) component of ambiguous wind field	REAL(4)	Array (nx,ny)	m/s	] -30, 30 [ <ul style="list-style-type: none"> <li>• for invalid grid point (land, ice, rain, bad retrieval) put: <i>v_amb</i>(<i>l:4, ix, iy</i>) = missing = + 9999</li> <li>• if <i>n_amb</i>(<i>ix, iy</i>) &lt; 4 put <i>v_amb</i>(<i>n_amb+1:4, ix, iy</i>) = missing = + 9999</li> </ul>

### OUTPUT:

Variable Name	Description	Type	Array (Length)	Range
<i>SELECTED</i>	Index for selected ambiguity	INTEGER	Array (nx,ny)	[1,4] for invalid grid point (land,

				ice, rain, bad retrieval) put <i>selected(ix,iy) = 1</i> [2, itermax]
<i>JITER</i>	Iteration Step	INTEGER(4)	Scalar	
<i>xgain</i>	Change in cost function between iteration step <i>jiter</i> and <i>jiter-1</i>	REAL(4)	Scalar	
<i>ireturn</i>	return code:  = +1: MF has converged, i.e. no fields were changed ( <b>good return</b> )  = +2: MF termination condition reached ( <b>good return</b> )  = -1: Maximum number of iterations exhausted ( <b>problematic return</b> )	INTEGER(4)		

The MF termination condition (*ireturn* = 2) is (c.f. CMIS Ocean ATBD by C. Smith, F. Wentz and T. Meissner, section 4.5.2.6):

$$|xgain| = \frac{\left| \sum_{ij} E_{ij}^{k_{\min}(N)} - \sum_{ij} E_{ij}^{k_{\min}(N-1)} \right|}{\sum_{ij} E_{ij}^{k_{\min}(N-1)}} < \varepsilon = 10^{-3} \text{ and } xgain < 0.$$

$E_{ij}^{k_{\min}(N)}$  denotes the minimum cost function at iteration step N (i.e. evaluated the ambiguity where it has its minimum value).

## GENERAL

### Major Subroutines

The code package consists of 4 major subroutines, which are to be called by the user:

- |                              |                                |
|------------------------------|--------------------------------|
| 3. <b>retrieve_sst</b>       | <i>SST RETRIEVAL</i>           |
| 4. <b>retrieve_wind_20km</b> | <i>20 km WIND RETRIEVAL</i>    |
| 5. <b>wind_stress</b>        | <i>Wind Stress Computation</i> |
| 6. <b>fdwindvec</b>          | <i>Wind Vector Retrieval</i>   |

### Dependencies

The wind vector retrieval **fdwindvec** uses a number of dependent subroutines and functions, which are not to be called by the user:

- fdwspd\_LR
- fdatm
- MLE
- phimin
- chi2\_w
- chi2
- find\_tbmodel
- find\_emiss
- find\_scat
- find\_stokes
- minimum\_search
- shuffle
- indx
- sort
- re\_arrange

### External Files

The following 6 external ASCII table files (contained in the folder *input\_tables*) are read in:

FILE NAME	FILE named in Program	SUBROUTINE	UNIT
<i>coeff_sst.lis</i>	file_name	retrieve_sst	3
<i>coeff_wind_20.lis</i>	file_name	retrieve_wind_20km	3
<i>coeff_wind.lis</i>	file_name	fdwspd_LR	3
<i>coeff_atm.lis</i>	file_name	fdatm	3
<i>emiss_ascii.lis</i>	emiss_file	find_tbmodel	21
<i>scat_ascii.lis</i>	scat_file	find_tbmodel	22

The file specification is done at the beginning of the corresponding subroutine, so that the user can easily adjust the PATH + FILE specification as it fits to user's system.

## SST RETRIEVAL CODE

### SUBROUTINE:

**retrieve\_sst ( TB, THT\_36, SST, IERR)**

### USAGE:

**CALL retrieve\_sst ( TB, THT\_36, SST, IERR)**

### DESCRIPTION:

This subroutine calculates the SST [in Celsius]. It is a regression using the brightness temperatures  $TB(i), i=1, 18$  [in Kelvin] of the 18 CMIS channels below 36 GHz.

The channel line up for  $TB$  is:

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Freq	6	6	10	10	10	10	18	18	18	18	18	18	23	23	36	36	36	36
POL	V	H	V	H	L	R	V	H	P	M	L	R	V	H	V	H	P	M

The form of the regression is

(c.f. : ATBD: OCEAN EDR ALGORITHM SUITE by C. Smith, F. Wentz and T. Meissner)

$$SST = c_0 + \sum_{i=1}^{18} c_i t_i + \sum_{i=1}^{18} c_i t_i^2$$

where:

$$t_i = T_{B_i} - 150K, \quad i \neq 13, 14 \text{ (all channels except 23 GHz channels)}$$

$$t_i = -\ln(290K - T_{B_i}), \quad i = 13, 14 \text{ (23 GHz channels)}$$

If the subroutine is called the first time, a table of regression coefficients *coeff\_sst* is read in from an external file *file\_name* which needs to be specified by the user (UNIT 3).

This table contains the regression coefficients for 5 different Earth Incidence Angles (EIA) at 36 GHz:

$THT0 - 1.0 \text{ deg}$ ,  $THT0 - 0.5 \text{ deg}$ ,  $THT0$ ,  $THT0 + 0.5 \text{ deg}$ ,  $THT0 + 1.0 \text{ deg}$

where  $THT0 = 55.789 \text{ deg}$  denotes the nominal EIA for 36 GHz.

The actual value for the 36 GHz EIA is called *THT\_36* [in deg]. It is required that *THT\_36* is within 1 deg of *THT0*. Two separate regressions are performed with the regression coefficients from the table that correspond to the EIA closest to *THT\_36*. The two results are then linearly interpolated to *THT\_36*.

**INPUT:**

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>TB</i>	Brightness Temperatures	REAL(4)	Vector (16)	Kelvin	$TB(13) < 290 K$ $TB(14) < 290 K$
<i>THT_36</i>	EIA at 36 GHz	REAL(4)	Scalar	deg	$THT0 - 1 \text{ deg} /$ $THT\_36 /$ $THT0 + 1 \text{ deg}$ where $THT0 = 55.789 \text{ deg}$

**OUTPUT:**

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>SST</i>	Sea Surface Temperature	REAL(4)	Scalar	Celsius	[-2,40]
<i>IERR</i>	Error Code (out of bound check for input variables)	INTEGER(4)	Scalar		

*IERR* = 0: normal return

*IERR* = -1:  $TB(13) > 290 K$  or  $TB(14) > 290 K$

*IERR* = +1:  $THT\_36 < THT0 - 1 \text{ deg}$

*IERR* = +2:  $THT\_36 > THT0 + 1 \text{ deg}$

**USER SPECIFIED:**

*file\_name*: external (file + path), which contains the table of regression coefficients *coeff\_sst* (5%37). The file is formatted, UNIT = 3.

# 20 km WIND SPEED RETRIEVAL CODE

## SUBROUTINE:

**retrieve\_wind\_20km ( TB, THT\_36, WIND\_20, IERR)**

## USAGE:

**CALL retrieve\_wind\_20km ( TB, THT\_36, WIND\_20, IERR)**

## DESCRIPTION:

This subroutine calculates the 20 km wind speed *WIND\_20* [in m/s]. It is a regression using the brightness temperatures  $TB(i), i=1,12$  [in Kelvin] of the 12 CMIS channels above 18 GHz and below 36 GHz.

The channel line up for *TB* is:

i	1	2	3	4	5	6	7	8	9	10	11	12
Freq	18	18	18	18	18	18	23	23	36	36	36	36
POL	V	H	P	M	L	R	V	H	V	H	P	M

The form of the regression is

(c.f. : ATBD: OCEAN EDR ALGORITHM SUITE by C. Smith, F. Wentz and T. Meissner)

$$W_{20km} = c_0 + \sum_{i=7}^{18} c_i t_i + \sum_{i=7}^{18} c_i t_i^2$$

where:

$$t_i = T_{B_{i-6}} - 150K, \quad i \neq 13,14 \text{ (all channels except 23 GHz channels)}$$

$$t_i = -\ln(290K - T_{B_{i-6}}), \quad i = 13,14 \text{ (23 GHz channels)}$$

If the subroutine is called the first time, a table of regression coefficients *coeff\_wind\_20* is read in from an external file *file\_name*, which needs to be, specified by the user (UNIT 3).

This table contains the regression coefficients for 5 different Earth Incidence Angles (EIA) at 36 GHz:

*THT0 - 1.0 deg, THT0 - 0.5 deg, THT0, THT0 + 0.5 deg, THT0 + 1.0 deg*

where *THT0 = 55.789 deg* denotes the nominal EIA for 36 GHz.

The actual value for the 36 GHz EIA is called *THT\_36* [in deg]. It is required that *THT\_36* is within 1 deg of *THT0*. Two separate regressions are performed with the regression coefficients from the table that correspond to the EIA closest to *THT\_36*. The two results are then linearly interpolated to *THT\_36*.

**INPUT:**

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>TB</i>	Brightness Temperatures	REAL(4)	Vector (12)	Kelvin	$TB(7) < 290 K$ $TB(8) < 290 K$
<i>THT_36</i>	EIA at 36 GHz	REAL(4)	Scalar	deg	$THT0 - 1 \text{ deg} /$ $THT\_36 /$ $THT0 + 1 \text{ deg}$ where $THT0 = 55.789 \text{ deg}$

**OUTPUT:**

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>WIND_20</i>	Sea Surface Temperature	REAL(4)	Scalar	m/s	
<i>IERR</i>	Error Code (out of bound check for input variables)	INTEGER(4)	Scalar		

*IERR* = 0: normal return

*IERR* = -1:  $TB(7) > 290 K$  or  $TB(8) > 290 K$

*IERR* = +1:  $THT\_36 < THT0 - 1 \text{ deg}$

*IERR* = +2:  $THT\_36 > THT0 + 1 \text{ deg}$

**USER SPECIFIED:**

*file\_name*: external (file + path), which contains the table of regression coefficients *coeff\_wind\_20* (5%37). The file is formatted, UNIT = 3.

# WIND STRESS COMPUTATION

## SUBROUTINE:

`wind_stress ( WIND, WS)`

## USAGE:

`CALL wind_stress ( WIND, WS)`

## DESCRIPTION:

This subroutine calculates the Wind Stress WS [in N m\*\*-2] for a given wind speed WIND [in m/s]. The coefficient of drag is tabulated for W between 0 and 25 m/s in increments of 1 m/s and linearly interpolated to the actual value of *WIND*.

For details see: ATBD: OCEAN EDR ALGORITHM SUITE by C. Smith, F. Wentz and T. Meissner (section 4.6)

## INPUT:

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>WIND</i>	Wind Speed	REAL(4)	Scalar	m/s	[0,25] m/s

## OUTPUT:

Variable Name	Description	Type	Array (Length)	Unit	Range
<i>WS</i>	Wind Stress	REAL(4)	Scalar	N/m**2	

# WIND VECTOR RETRIEVAL CODE

## SUBROUTINE:

**fdwindvec ( SST,THT,TB, N\_AMBIG,WIND\_AMBIG,PHI\_AMBIG,CHISQ, IERR)**

## USAGE:

**CALL fdwindvec ( SST,THT,TB, N\_AMBIG,WIND\_AMBIG,PHI\_AMBIG,CHISQ, IERR)**

## DESCRIPTION:

### WIND VECTOR RETRIEVAL:

1. find first guess for wind speed W0 by regression
2. regression for atmospheric parameters TBUP, TBDW, TRAN
3. MLE MINIMIZATION OF SOS (CHISQ) with W0 as starting value and model function evaluated for SST, TBUP, TBDW and TRAN.

The routine returns the number of ambiguities and up to 4 ambiguous solutions for the wind vector together with the values for chisq. It uses the brightness temperatures  $TB(i), i=1, 16$  [in Kelvin] of the 12 CMIS channels above 10 GHz and below 36 GHz.

The channel line up for  $TB$  is:

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Freq	10	10	10	10	18	18	18	18	18	18	23	23	36	36	36	36
POL	V	H	L	R	V	H	P	M	L	R	V	H	V	H	P	M

For details see: ATBD: OCEAN EDR ALGORITHM SUITE by C. Smith, F. Wentz and T. Meissner, section 4.5.

## INPUT:

Variable Name	Description	Type	Array (Length)	Unit	Range
SST	Sea Surface Temperature	REAL(4)	Scalar	Celsius	[-2, 40]
THT	EIA at 5 CMIS frequencies: 1 = 6 GHz (not used) 2 = 10 GHz 3 = 18 GHz 4 = 23 GHz 5 = 36 GHz	REAL(4)	Vector (5)	deg	50 deg < THT(i) < 60 deg THT0 - 1 deg [ THT(5) / THT0 + 1 deg where THT0 = 55.789 deg
TB	Brightness Temperatures	REAL(4)	Vector (16)	Kelvin	TB(11) < 290 K TB(12) < 290 K

## OUTPUT:

Variable Name	Description	Type	Array (Length)	Unit	Range
N_AMBIG	Number of Ambiguities	INTEGER(4)	Scalar		[1,4]
WIND_AMBIG	Wind Speed of	REAL(4)	Vector	m/s	]1,26[

	Ambiguities (ordered by ascending SOS)		(4)		
<i>PHI_AMBIG</i>	Relative Wind Direction of Ambiguities (ordered by ascending SOS)	REAL(4)	Vector (4)	deg	[0,360[
<i>CHISQ</i>	SOS from MLE (ascending)	REAL(4)	Vector (4)	Kelvin	> 0
<i>IERR</i>	Error Code	INTEGER(4)	Scalar		

If  $N\_AMBIG < 4$ , then the remaining entries in *WIND\_AMBIG*, *PHI\_AMBIG* and *CHISQ* are filled with +9999 (missing value).

### ERROR CODES:

<b>IERR</b>	<b>EXPLANATION</b>
0	normal return
1	the first guess wind speed W0 is out of bound; needs to be within ]1,26[
2	THT out of bound; requires: 50 deg < THT(i) < 60 deg and THT0 – 1 deg [ THT(5) [ THT0 + 1 deg where THT0 = 55.789 deg
3	TRAN out of bound; needs to be within [0,1].
4	Failure in minimum search: n_ambig = 0 No minimum found.
5	Failure in minimum search: minimum cannot be cornered. SOS(W0) needs to be smaller than both SOS(0) and SOS(26).
6	Failure in minimum search: accuracy goal $ y_1 - y_4  < \epsilon \frac{ y_1 + y_4 }{2}, \quad \epsilon = 10^{-3}$ not achieved after NMAX = 50 iterations. ( $[y_1, y_4]$ is the interval cornering the minimum).
7	Failure in minimum search: n_ambig $\mu$ 5 Too many ambiguities.
8	TB(11) > 290K or TB(12) > 290K.
9	SST out of bound Needs to be within [2,40].

### USER SPECIFIED:

<b>FILE NAME</b>	<b>FILE named in Program</b>	<b>SUBROUTINE</b>	<b>UNIT</b>
<i>coeff_wind.lis</i>	file_name	fdwspd_LR	3
<i>coeff_atm.lis</i>	file_name	fdatm	3
<i>emiss_ascii.lis</i>	emiss_file	find_tbmodel	21
<i>scat_ascii.lis</i>	scat_file	find_tbmodel	22

# Upper Atmospheric Vertical Temperature Profile

## Executable Program:

Test\_dop\_geomag

## Description :

uavtp generates upper atmospheric vertical temperature profile from retrieval products.

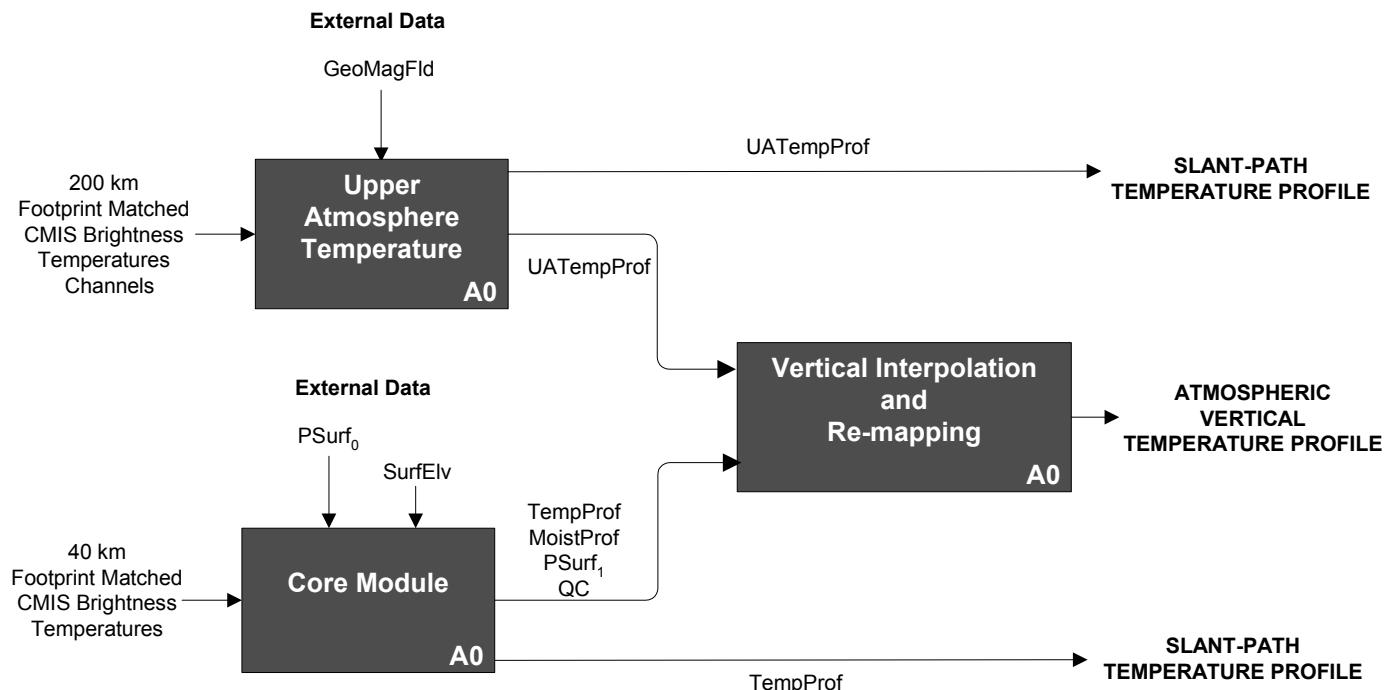
## Syntax

Seen the README in EDR/uatp

## Source Code Location:

EDR/uatp

## Function Flow Diagram:



## KEY

### Products from CMIS Data

Input	Description	Source
TempProf	Atmospheric Temperature Profile	Core Module
MoistProf	Atmospheric Moisture Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
UATempProf	Upper Atmosphere Temperature Profile	Upper Atmosphere Algorithm

### Flags from CMIS Algorithm Output

Input	Description	Source
QC	Output Production Mode	Core Module

### Data from Sources External to CMIS

Input	Description	Example Source
PSurf <sub>0</sub>	Surface Pressure	NCEP or FNMOC NOGAPS
SurfElev	Surface Elevation	DTED
GeoMagFld	Geomagnetic Field Model	IGRF or DoD

# Precipitation Type/Rate

Executable Program:

cmis\_precip\_test

**Description :**

Precipitation type/rate algorithm generates precipitation rate.

**Syntax:**

**Compile:**

C89 -o cmis\_precip\_test cmis\*.c array\_utils.c -lm

Where c89 is an ANSI C compiler.

**Run:**

cmis\_precip\_test 99010300 080\_010\_00\_00\_00\_00

The first argument is a date/time string ; the second is a string identifying the RT mode configuration.

**Inputs:**

99010300\_006625\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_006625\_080\_010\_00\_00\_00\_1111111.cmis  
99010300\_010650\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_010650\_080\_010\_00\_00\_00\_1111111.cmis  
99010300\_018700\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_018700\_080\_010\_00\_00\_00\_1111111.cmis  
99010300\_023800\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_023800\_080\_010\_00\_00\_00\_1111111.cmis  
99010300\_036500\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_036500\_080\_010\_00\_00\_00\_1111111.cmis  
99010300\_089000\_080\_010\_00\_00\_00\_1100000.cmis  
99010300\_089000\_080\_010\_00\_00\_00\_1111111.cmis

99010300\_080\_010\_00\_00\_00\_00.rr

The files with the .cmis suffix are simulated CMIS swath data arrays.

The first line of each file gives the dimensions of the array, with the first value being the number of pixels per scan and the second being the number of scans for the particular frequency in question (given in MHz in the second group of digits in the file name). All subsequent lines in each file are pixel data: the first two values give the pixel and scan index values (counting from zero), the next two values give the geographic displacements in km from the initial satellite subpoint (for plotting purposes), and the final two values on a line give the vertically and horizontally polarized simulated

brightness temperatures. Files named \*1111111.cmis contain simulated brightness temperatures that take into account all atmospheric constituents and hydrometeors, including air, vapor, cloud water, rain water, graupel, snow, and ice. Files name \*1100000.cmis represent simulations with all condensed water removed (i.e., these are "cloud-free" simulations).

### **Outputs:**

99010300\_080\_010\_00\_00\_00.rret

Each line of this file contains data for one reporting cell: x,y coordinates (for plotting purposes), "true" precipitation rate, "true" liquid-only precipitation rate, and retrieved precipitation rate (all in mm/hr).

### **Source Code Location:**

EDR/precip

See README file under this directory for more detail information.

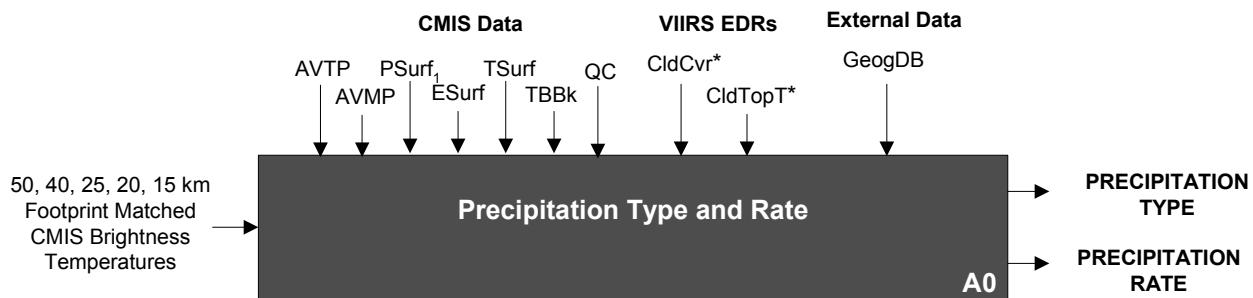
### **Source Codes:**

cmis\_precip\_test.c  
cmis\_precip\_core.c  
cmis\_precip\_globals.c

cmis\_precip.c  
cmis\_precip\_debug.c  
cmis\_precip\_init.c  
cmis\_precip\_main.c  
cmis\_precip\_preproc.c  
cmis\_precip.h  
cmis\_precip\_coeff.h  
cmis\_precip\_core.h

cmis\_precip\_debug.h  
cmis\_precip\_globals.h  
cmis\_precip\_init.h  
cmis\_precip\_main.h  
cmis\_precip\_preproc.h  
cmis\_precip\_sensor.h

## Function Flow Diagram:



KEY		
<b>Products from CMIS Data</b>		
Input	Description	Source
AVTP	Atmospheric Temperature Profile	Core Module
AVMP	Atmospheric Moisture Profile	Core Module
PSurf <sub>1</sub>	Surface Pressure	Core Module
ESurf	Surface Emissivity	Core Module
TSurf	Surface Temperature	Core Module
<b>Flags from CMIS Algorithm Output</b>		
Input	Description	Source
QC	Output Production Mode	Core Module
<b>Data from Sources External to CMIS</b>		
Input	Description	Example Source
GeogDB	Geography Database	DTED
CldCvr*	Cloud Cover Fraction	VIIRS Cloud Cover EDR
CldTopT*	Cloud Top Temperature	VIIRS Cloud Top Temperature EDR

\* = Optional Data Requirement

This page intentionally left blank.